制約つきグラフ探索を実現する
異種データベース統合技術に関する研究

鈴木　源吾

電気通信大学大学院
情報システム学研究科
社会知能情報学専攻

博士（工学）学位申請論文

2013 年 6 月

制約つきグラフ探索を実現する
異種データベース統合技術に関する研究

博士論文審査委員会

主査　大須賀　昭彦　教授
委員　大森　匡　教授
委員　鬼塚　真　教授
委員　田中　健次　教授
委員　田野　俊一　教授

Heterogeneous Database Integration Method for Constrained Graph Search

Gengo Suzuki

Abstract

Graph database services such as traffic search, map search, or SNS search have recently been getting more popularities. In the future integrating existing database services and graph database services will have more importance. For realizing such integration we have to resolve heterogeneities of data expression and search ability levels of databases.

We propose new heterogeneous database integration method for graph database search, which includes constraining condition for properties of nodes and edges in graphs. We call such graph database search "constrained graph search". We applied this method for practical "time constrained trip planning" problem, so our method is very effective for distributed graph databases.

Our method is composed by two sub-methods,

- 1st Sub method: Schema integration method using conceptual graphs
- 2nd Sub method: Data integration method with dynamic heterogeneity resolution

We can integrate several database schemas includes graph databases using the first sub method, and can get several metadata-dictionaries about concepts in several databases. The second sub method uses those metadata-dictionaries to resolve heterogeneities after users give retrieval requests to our system. One advantage of our method is dynamical heterogeneous resolution, so mapping definitions between databases are stable when existing database schema changes. Another advantage is query optimization under the variation of information source abilities for graph search. Our method can push-down sub queries to information sources, and query performance can be improved.

We showed our method's advantages by applying to time constrained trip planning search we proposed. This time constrained trip planning search is one kind of shortest path problem to search paths which include node in service of some category. A typical example is to find paths from office to home via some restaurant under constrains of time services. We constructed distributed graph databases environment and evaluate the method.

制約つきグラフ探索を実現する異種データベース統合技術に関する研究

鈴木　源吾

概 要

交通検索・地図検索・SNS 検索等でグラフ情報を活用するサービスが，近年多く開発され実用化されている．今後は，情報の表現形式や探索能力に異種性のあるグラフデータベースと既存のデータベースや Web サービスを組み合わせた統合検索が重要になると考えられる．本研究では，異種データベース統合技術をグラフデータ操作に拡張し，グラフ探索とノード・エッジのプロパティに対する制約条件とを組み合わせる，制約つきグラフ探索を可能とする手法を提案する．さらに，時間が限定されたサービスが実施されているノードを経由する最短経路問題である，時間制約つき寄り道経路探索を提案し，異種分散データベース環境において本手法を適用し有効性を示した．時間制約つき寄り道探索は，道路・鉄道等の交通路に関するデータベースと，店・開店時間等のサービスに関するデータベースは分散し独立して構築されている場合が多く，また，交通路情報が比較的安定であることに比べ，サービス情報は追加や変更が多いという特長を持つために，本手法が有効な典型的な例となっている．

提案手法は以下の２つの技術により構成される．

- 技術 1：概念グラフを利用したスキーマ統合技術
- 技術 2：動的に異種性解消するデータ統合検索技術

技術 1 を利用し，グラフデータベースを含む複数のデータベーススキーマを統合し，対応関係のメタデータを構築する．技術 2 を利用し，検索要求時にメタデータを探索し，命名・構造・表現の異種性を解消し，情報源に対する問合せの組合せを作成し，統合検索を実行する．検索時に検索対象とその異種性解消を決定する動的な側面と，情報源の能力に応じてグラフ探索を最適化できる点に特長がある．情報源の能力にばらつきがあるときに，グラフデータベースへの探索の一部を，情報源側に適切にプッシュダウンすることで高速化することが可能である．

技術 1 で用いている概念グラフは，ER モデル等と比較して，データモデル構成要素が少ないため，同じ意味を異なる構造で表現する異種性である構造異種性を発生しないことに特長があるデータモデルである．そこで，ER モデル等によって表現されたスキーマを一旦，概念グラフに変換してから比較することにより，構造異種性を回避することができる．さらにデータ項目が一定のルールで標準的に構造化されている場合（Durell-関根の標準化ルール），概念グラフに変換するとき，データ項目を複数の概念に分解し，マッチングを増すことができる特長がある．この手法は，特に日本語の項目名のように複合語が多い場合に有効である．ER モデルとグラフデータベースから概念グラフへ変換する手法と類似度計算を含む統合手法を明らかにした．グラフデータベースの機能もスキーマとしてモデル化する．本手法の過程において，データ項目と概念の対応関係・概念同士のつながり・概念とデータ項目の表現形式のメタデータを得ることができる．

技術 2 の動的に異種性解消するデータ統合検索技術は，分散されたグラフ探索処理機能を検索時に動的に組み合わせ，異種性解消するデータ統合技術であり，関係データモデルを用いている． Web 情報源の技術を用いグラフデータベースに対する制約つき寄り道探索機能を仮想的な表としてデータ統合可能にする．さらに，能力にばらつきのあるグラフデータベース情報源に対して，適切にグラフ探索のプッシュダウンを行うために，グラフ探索能力を階層化し最適化する手法を提案した．グラフ探索能力が高くなるほど，多くのデータ項目を利用する点に着目し，適切な検索候補の生成法を明らかにした．

時間制約つき寄り道経路探索に対して本手法を適用するために，まず探索を定義・定式化し，単一データベースに対する導出法を明らかにした．時間制約のない寄り道探索結果を求める既存手法である逐次拡大法の結果に対して，制約条件をチェックする手法である基本導出法を示す．さらに，制約条件のチェックをできるだけ早期に行い探索の枝刈りを行う動的導出法を提案した．単一データベース構成では，動的導出法が有利であることを示した．

異種分散データベース環境に拡張するために，時間制約つき寄り道探索の処理を分解し，情報源の能力のバリエーションを定義した．交通情報とサービス情報のデータ分散形態を整理し，基本導出法・動的導出法を用いた場合，本手法を適用してプッシュダウンを含む問合せ戦略を得ることができる．その問合せ戦略を，分散グラフデータベース環境における実験で評価し，プッシュダウンが有効になる基本導出法の有効範囲が広いことを示した．

本研究により，制約つきグラフ探索を異種分散データベース構成で実現する基本技術が確立した．今後の課題としては，他のグラフ探索処理（例：POI 複数化）へ適用し，関係モデルの拡張方式の限界を見極めることと，探索以外のグラフ処理への適用がある．

目 次

図 目 次

表 目 次

第1章　はじめに

交通検索・地図検索・SNS 検索等でグラフ情報を活用するサービスが，近年多く開発され実用化されている．交通・地図検索としては駅探のような鉄道路線検索やバス路線の検索サービス，Google Map による道路網のナビゲーションサービス等が Web 上で容易に利用可能となっている．また，最近公開された Facebook のグラフ検索では，人の属性や関係性（例：サイクリングが好きで近くに住んでいる友達）を表す問合せによって SNS グラフを検索することができる．これらのサービスは，グラフ情報を扱う専用のデータベースであるグラフデータベース [A3][A75] によって実現されるようになりつつある．グラフデータベースでは，情報はノードとそれをつなぐエッジ，それぞれに付与されるプロパティでモデル化され，最短経路探索等のグラフ探索，SPARQL による知識検索，パターンマッチ等のグラフ独自処理を高速に実行することができる．

現在のグラフ情報利用サービスは，最短経路探索等の比較的単純なグラフ探索技術を利用することが多いが，今後はグラフに対して様々な条件を指定した経路探索の必要性が増えると考えられる．図 1.1 は，SNS に対する条件探索の例であり，「ある人と，2 ホップ以内のつながりで趣味が音楽の人」までの経路を求める探索である．この場合，ノードのプロパティである「趣味」に対して，制約条件をつけて探索を行なっている．図 1.2 は，交通網と店とのつながりに関するグラフの探索であり，ノードのプロパティに対する条件として，店のカテゴリを指定しており，店の開店時間に対する時間制約も指定している（この例は，時間制約つき寄り道探索と呼んでおり，本研究の主要な応用として後に詳述する）．

このような，グラフのノードとエッジのプロパティに対して条件を指定するグラフ経路探索を，「制約つきグラフ探索」と呼ぶこととする．制約つきグラフ探索は，本論文で定義した用語であり，過去に一般的に用いられているものではない．その定義は次章で示しているが，その基本的特徴は以下の 3 つである．

- 対象データはグラフ：座標の集まりではなく，ノードとエッジであり，軌跡探索などの地理情報的なモデルではない．
- グラフ上のコストで評価：エッジのプロパティとしてコスト（時間・距離等）を与える．ユークリッド距離（座標間の距離）による評価ではない．

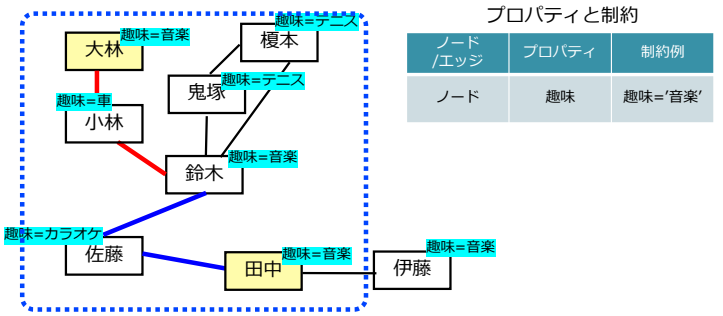


図 1.1: 制約つきグラフ探索：SNS の条件探索

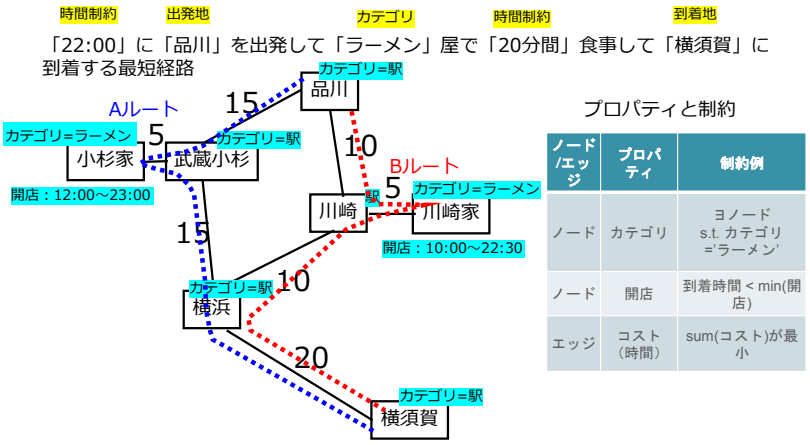


図 1.2: 制約つきグラフ探索：時間制約つき寄り道探索

- 探索は経路探索：プロパティで指定した制約を満たす「経路」を求める．グラフのパターンによる探索（例：化学構造探索）は対象外である．

制約つきグラフ探索では，グラフのノード・エッジ情報と制約条件のプロパティは，複数の情報源にまたがることが多い．例えば，レストラン情報と交通情報，SNS のつながりとレビューサイトのカテゴリ情報，ネット上の RDF 情報と社内 DB 等があげられる．これらの情報源を組み合わせることによって，制約つきグラフ探索を実現する必要があるが，既存の情報源とグラフデータベースは，そもそも組み合わせて連携することを前提に設計されていない．よって，これらの情報源・データベースは分散して様々な異種性を持っていることがほとんどである．

そこで本研究では以下を研究目標とする，そのイメージを図 1.3 に示す．

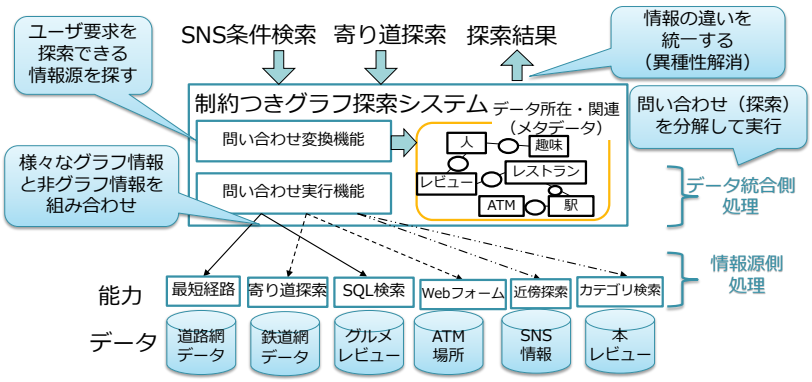


図 1.3: 本研究の目標

次世代のグラフデータベース活用サービスの確立に向けた，
制約つきグラフ探索の異種分散データベース環境での実現

様々なグラフデータと既存情報源が分散して存在するときに，それらを統合検索できる制約つきグラフ探索システムが，ユーザからの SNS 条件探索，寄り道探索等の様々な問い合わせ要求に対し，探索結果を求めることができる．制約つきグラフ探索システムは，ユーザ要求から探索できる情報源を探して特定し（問い合わせ変換機能），問い合わせを分解して，様々なグラフ情報と非グラフ情報に対して探索・問い合わせ要求を発行し，その結果を組み合わせ，情報の違いを統一しユーザに返却する（問い合わせ実行機能）．

この研究目標を実現するために，以下の 2 つの課題がある．

課題 1：スキーマ間の異種性解消と統合　独立して設計された情報源には，情報の構造・表現形式・命名に違い（異種性）がある．その違いを解消しつつデータ統合検索を実現する必要がある．その異種性を解消するための，複数の情報源の突き合わせ（スキーマ統合）には多大な稼働がかかり，完全自動化が困難であることが知られている [A6]．その稼働削減には，適切なモデル化・支援方式・ツールがあることが望ましい．しかし，グラフデータベースは新しい情報源であり，そのスキーマ統合手法は未確立である．

課題 2：情報源能力のばらつきに対応した問い合わせ処理　グラフ利用サービスが発展途上であるため，グラフデータベースの機能はまだ確立しておらず流動的である．よって，グラフデータベースには，RDB における SQL のような標準言語が存在しない，また，Web 経由でサービスが提供されるケースが多いため，そのポリシーによってすべてのグラフ探

索 API が公開されるとも限らない．これらの理由から，グラフデータベースを利用する情報源には，グラフ探索できる能力にばらつきがある．よって，制約つきグラフ探索システムは，このようなばらつきを考慮しつつユーザからの要求を適切に，情報源による探索処理（情報源側処理）と制約つきグラフ探索システム内部の処理（データ統合側処理）に分解して，処理する必要がある．グラフ探索処理は，頻繁にノード・エッジにアクセスすることが一般的であるために，できるだけ情報源側へ処理を移譲するような最適化が望ましい，今後，グラフ探索処理の分担を説明するときに，この「情報源側」「データ統合側」という用語を用いることとする．また，グラフ利用サービスは，探索サービスの内容や利用する項目の頻繁な変更が多いため，その能力のばらつきも変化が大きい．よって，スキーマ変更への追随が容易な方式が望ましい．

これらの課題の解決のために，本論文では，2 つの手法を提案する．

まず，データ統合に必要な，データ項目と概念の対応関係・概念同士のつながり・概念とデータ項目の表現形式対応のメタデータを効率的に構築するために，概念グラフを用いたスキーマ統合技術 [B76][B80] を確立した（手法 1）．このスキーマ統合技術は，グラフデータベースを含む様々な情報源スキーマを「概念グラフ」と呼ばれるモデルに変換して異種性を解消することが特徴である．概念グラフは，ER モデル等と比較して，データモデル構成要素が少ないため，同じ意味を異なる構造で表現する異種性である構造異種性を回避しやすいことが特長である．また，データ項目が一定のルールで標準的に構造化されている場合に，データ項目を複数の概念に分解し，マッチングを増すことができる．概念間の類似度計算法を組み込んだスキーマ統合支援ツールを実現することにより，スキーマ統合稼働の削減の可能性を示し，課題 1 を解決した．

次に，動的に異種性解消するデータ統合検索技術（手法 2）[B81][B82] を確立した．利用側の要求とデータベースの構造を事前に SQL 等の言語によって固定的に結びつけるのではなく，検索要求時に情報源の所在や表現形式等のメタデータを探索して，検索要求に答えることのできる情報源への問合せの組合せ（問合せ候補と呼ぶ）を生成し，動的に命名・構造・表現の異種性を解消する．そのイメージを図 1.4 に示す．メタデータ層で情報源を抽象化することによって動的な検索を実現ため，サービス変更に柔軟な動的な特性を持つ．問合せ候補生成には，既存の情報源のスキーマを統合した統合概念グラフを利用する．ユーザの要求を，統合概念グラフのどの概念に対応するかを特定し，その概念間のつながりを探索する．探索結果と概念とデータベースの対応情報を利用することによって情報源への問合せを生成することができる．

さらに，このデータ統合検索技術では，制約つきグラフ探索の能力を階層的にモデル化することにより，ばらつきのある情報源が処理できる能力を考

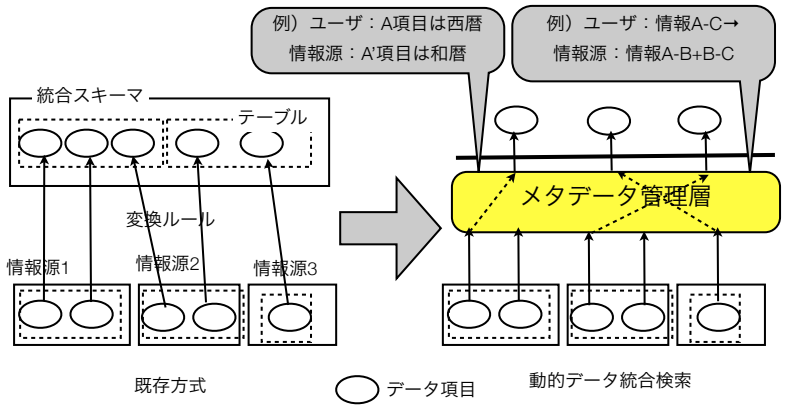


図 1.4: 動的なデータ統合検索のイメージ

慮し，情報源側に適切なプッシュダウンを行うことが可能である．制約つきグラフ探索の複雑さが増すほど，多くのデータ項目を利用する点に着目し，適切な検索候補の生成法を明らかにした．この技術により課題 2 が解決できる．

これら 2 つの提案手法の有効性を示すために，制約つきグラフ探索の一例として時間制約つき寄り道探索を提案・定式化し，提案手法を適用した．時間制約つき寄り道探索は，利用者の時間制約（例：門限）とサービス側の時間制約（例：開店時間）を満たす条件で，指定したサービス（例：店の開店）を持つノードを経由する最短経路を求める探索である [B79][A65]，道路・鉄道等の交通路に関するデータベースと，店・開店時間等のサービスに関するデータベースは分散し独立して構築されている場合が多く，また，交通路情報が比較的安定であることに比べ，サービス情報は追加や変更が多いという特長を持つために，提案手法が有効な典型的な例となっている．

まず，単一データベースにおける探索結果の導出手法を明らかにした．時間制約のない寄り道探索結果を求める既存手法である逐次拡大法の結果に対して，制約条件をチェックする手法である基本導出法と，制約条件のチェックをできるだけ早期に行い，探索の枝刈りを行う動的導出法を確立した．さらに単一グラフデータベース構成における評価を行い，動的導出法が有利であることを示した．

異種分散データベース環境に拡張するために，時間制約つき寄り道探索の処理を分解し，情報源の能力のバリエーションを定義した．交通情報とサービス情報のデータ分散形態を整理し，基本導出法・動的導出法を用いた場合，本手法を適用してプッシュダウンを含む問合せ戦略を得ることができる．その問合せ戦略を，分散グラフデータベース環境における実験で評価し，プッシュダウンが有効になる基本導出法の有効範囲が動的導出法に比べて広いことを示した．

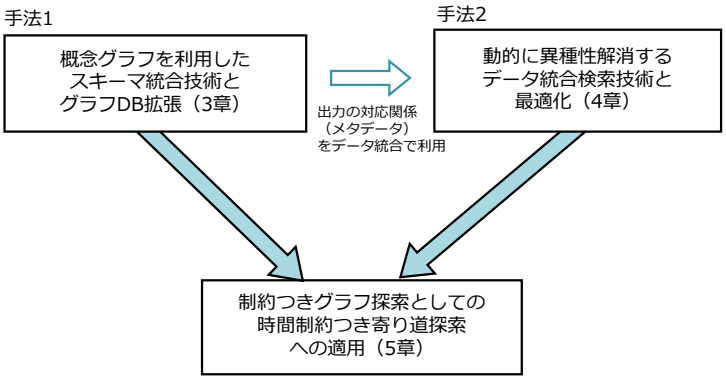


図 1.5: 本論文の構成

本論文の構成を図 1.5 に示す．まず，第 2 章にて，制約つきグラフ探索を定義し，本研究の背景と位置づけを，グラフ探索技術と既存の異種分散データベース技術との関連も含めて述べる．3 章では，手法 1 の概念グラフ変換を利用したスキーマ統合技術について述べる．4 章では，手法 2 の動的に異種性解消するデータ統合検索技術と最適化について述べる．手法 1 のスキーマ統合の過程で作成された，スキーマ間の対応関係のメタデータを手法 2 で利用するという関係となる．5 章で，制約つきグラフ探索の例である，時間制約寄り道探索への適用について述べる．まず時間制約寄り道探索を定義し，単一データベースにおける解の導出法を示す．次に，異種分散データベース環境における提案手法の適用について述べ，その適用の評価結果を示す．6 章で本手法の適用範囲と効果を考察し，7 章で研究の到達点と今後の課題を結論として述べる．

第2章　制約つきグラフ探索の定義と本研究の位置づけ

本章では，本研究で取り組む問題である制約つきグラフ探索を定義し，そのグラフ探索技術における位置づけを示す．そして，それを異種分散データベース環境で実現することの位置づけを，異種分散データベース統合技術の俯瞰とともに示す．

2.1　制約つきグラフ探索

2.1.1　制約つきグラフ探索の定義

グラフに関する用語について説明する．本研究では，グラフの構成要素の用語はノードとエッジで基本的には一貫して使用することとする．ただし，グラフ理論ではグラフを Vertex と Edge で表すことが通例であるため，数式で表す場合は，$G = (V, E)$ 等の慣例的な記法を用い，ノードと頂点は同義であるとみなす．

まず，制約つきグラフ探索において，制約を指定するためにプロパティつきグラフの定義を行い，プロパティつきグラフに対して制約条件を指定した探索を行う制約つきグラフ探索を定義する．

定義 1 (プロパティつきグラフ) グラフ $G = (V, E)$ に対して，以下の定義によるプロパティ P が与えられているとき，G をプロパティつきグラフと呼び，G_P の記法で表す．

プロパティ P とは，ノード（頂点）またはエッジから，文字・数値への関数の集合であると定義し，ノードに関するプロパティをノードプロパティ，エッジに関するプロパティをエッジプロパティと呼ぶ．ノード・エッジに対するプロパティ関数の数（関数集合の要素数）をそれぞれ n, m とするとき以下の記法によりノード（エッジ）プロパティを記述することとする．

$$P^V = \{p_1^V, p_2^V, ..., p_n^V\}$$

$$P^E = \{p_1^E, p_2^E, ..., p_m^E\}$$

例 **1** (プロパティつきグラフ) ノードが駅の集合であり，プロパティとして路線・緯度・経度が与えられ，エッジが駅間の移動を表し，プロパティとして距離と移動時間が与えられたプロパティグラフは以下のように表現される（上記定義の $n = 3$, $m = 2$ の例である）.

$$\text{駅グラフ} = (\text{駅}, \text{駅間移動})$$

$$P^{\text{駅}} = \{line(), latitude(), longitude()\}$$

$$P^{\text{駅間移動}} = \{length(), time\_cost()\}$$

定義 **2** (制約つきグラフ探索) プロパティつきグラフ $G_P = (V_{P^V}, E_{P^E})$ に対して，ノード v_s（探索開始点）とノード v_e（探索終了点）を結び，以下に定義するプロパティに関する結果制約条件を満たすパスの集合を返却する探索を制約つきグラフ探索と定義する.

結果制約条件とは，結果パスが満たすべき，ノードプロパティとエッジプロパティ（の部分集合）を利用した数式及び論理式で規定される条件であり，以下の式で表す．結果制約条件で利用されるノード・エッジプロパティ集合を，それぞれ $\{p^V_{i_1}, p^V_{i_2}, ..., p^V_{i_{n'}}\}(n' \leq n)$，$\{p^E_{j_1}, p^E_{j_2}, ..., p^E_{j_{m'}}\}(m' \leq m)$ とする.

$$C(p^V_{i_1}, p^V_{i_2}, ..., p^V_{i_{n'}}, p^E_{j_1}, p^E_{j_2}, ..., p^E_{j_{m'}})$$

結果制約条件の中では，結果の順序を規定するための特別な関数 $order()$ を利用できるとする．$order()$ は，結果パス全体，特定の結果パス，順序を判定するためのプロパティを用いた数式，の 3 つの引数を持ち，特定の結果パスの結果パス全体における順位を返却する.

例 **2** (制約つきグラフ探索：ダイクストラ探索) ダイクストラ探索は制約つきグラフ探索の一種である．以下のように表現される．$G_P = (V_{P^V}, E_{P^E})$ に対し，エッジプロパティ $P^E = \{cost()\}$ のみが与えられる（$m = 1$）．結果制約条件 C は以下で定義される.

$$C(cost()) : order(path, all\_path, \sum_{e \in path} cost(e)) = 1$$

この定義は結果パスの $cost()$ の合計が全パス中で最小であることを表す.

例 **3** (制約つきグラフ探索：寄り道探索) 寄り道探索（詳細は 5 章にて述べる）は制約つきグラフ探索の一種である．以下のように表現される．$G_P = (V_{P^V}, E_{P^E})$ に対し，ノードプロパティ $P^V = \{category()\}$ エッジプロパティ $P^E = \{cost()\}$ が与えられる（$n = 1, m = 1$）．結果制約条件 C は以下で定義される.

$$C(category(), cost()) : \exists v, category(v) = \text{ラーメン屋}$$

$$\wedge order(path, all\_path, \sum_{e \in path} cost(e)) = 1$$

この定義はパスのどこかでラーメン屋を経由し，結果パスの $cost()$ の合計が全パス中で最小となる制約条件を表す．

制約つきグラフ探索は，結果制約条件 C に含まれるプロパティの数が増えるほど応用的で複雑な探索となる傾向となる．

2.1.2 制約つきグラフ探索の位置づけ

制約つきグラフ探索は，開始ノード・終了ノードを指定してパスを求めるグラフ探索と，ノードとエッジに対するプロパティに関する制約条件指定を組み合わせた探索である．これは筆者らが独自に定義しており，必ずしも一般的な用語ではない．制約つきグラフ探索は，既存研究においてはグラフ探索の応用問題として捉えられ，個別の問題毎に探索の解法が示されていて，一つのカテゴリとしてまとめて議論されることはなかった．次節で説明する旅行計画問合せはその典型的な例である．しかし，分散されたグラフデータベースに対する問合せ最適化を考えるときに，制約や探索のプッシュダウンを行うために，制約つきグラフ探索を一つのカテゴリとして捉え，課題と解決法を検討することが有効であることを，本研究は示している．

本研究では，グラフデータベースを用いた制約つきグラフ探索を実現するために，最終的には探索機能を仮想的な表としてモデル化する．プロパティに対する制約は，仮想的な表に含まれるデータ項目の検索条件に変換される．ノードとエッジを個別に指定することはできるが，グラフの構造やパターンを細かく指定した問合せを行うことはできない．例えば，文献 [A3] は主に化学式を想定応用として，グラフ構造のマッチング検索を行っているが，本研究は応用できない．さらに，グラフ探索の問題としては，入力の数値・文字・部分グラフ等の条件による真偽判定により結果を返す exact match と，何らかの類似度指標を設けその高さにより結果をランキングして返す similarity match に分類される．similarity match では入力としてグラフのパターンや構造を指定する場合が多いため，本研究を有効に利用することはできない．基本的には exact match が本研究の適用対象となる．

2.1.3 旅行計画問合せ技術との関係

制約つきグラフ探索において，グラフが地図や交通網を表し，結果制約条件が希望する滞在箇所や，滞在時間による制約等である場合，その探索は，旅行計画問合せ（Trip Planning Query）の一種であると言える．旅行計画問合せとは，例えば，交通の乗り換え検索 [A63] のように，出発地・到着地や（複数の場合もある）経由地やそのカテゴリー等を与えられ，適切な交通ルート

表 2.1: 旅行計画問合せに関する既存研究

観点＼手法	寄り道探索	OSR探索	場所による軌跡検索
訪問先の数	1	複数	複数
問合せ対象の形式	グラフ（ノードとエッジ）	座標の集まり	座標の集まり
評価する距離	グラフ上の距離	ユークリッド距離	ユークリッド距離
保持する情報の粒度	グラフ全体	座標の集まりの全体	座標の組（軌跡）の集まり
問合せに指定する条件	ノード・エッジのプロパティ	座標のプロパティの列	座標の集まり

や移動時間・滞在時間等の旅行計画を作成するような問合せを指す．本研究でとりあげている寄り道探索はその一種である．旅行計画問合せは，本研究の重要な応用先であると言え，関連が深いため，以下で説明する．

旅行計画問合せに関する研究は近年盛んに行われており，最短経路探索などを求めるグラフアルゴリズム [A62][A60][A61]，一定の制約のもとでタスクを最適に配置するスケジューリングアルゴリズム [A72] 等に関連が深く，近年はインデックス技術を用いて探索を効率化するなどのデータベース技術を活用する研究も進みつつある．代表的な既存研究として，寄り道探索 [A65]・Optimal Sequential Route 探索 [A66]・場所による軌跡探索 [A14] について説明する（3 手法の比較を図 2.1 に示す）．

[A65] における寄り道探索とは，出発点と到着点と経由地（経由地は利用者が興味のある場所という意味から POI（Point of interest）と呼ばれる）のカテゴリを 1 つ指定して，その経路を求める探索である．経由地が 1 つであるという制約はあるが，前処理によるインデックス構築等を使わずに，グラフ上の距離に基づいた最適解を求められることが特長である．

Optimal Sequential Route（OSR）探索は，出発点を指定し，順序も含めて指定した複数のカテゴリの経由地をたどる探索である．例えば，横浜を出発して，ATM でお金をおろし，お土産屋でお土産を買って，最後に中華料理屋に食事するルートを求める探索である．[A66] は，事前に探索用のインデックスを構築することにより，ユークリッド距離に基づいた準最適解を求める方式を示している．

場所による軌跡探索とは，旅行者等が訪れた座標の集まり（軌跡）に関するデータベースがあるときに，ユーザが指定する複数の場所に関する条件から，類似する軌跡を探索する技術である [A14]．

旅行計画問合せは，制約つきグラフ探索技術の有力な適用分野であるが，すべてを制約つきグラフ探索技術で実現できるわけではない．旅行行計画問

合せには 2 つのアプローチがある．問合せ対象の地図等をグラフデータでモデル化しグラフ上の距離によって旅行計画の妥当性を判断するグラフ的アプローチと，座標の集合としてモデル化しユークリッド距離等で旅行計画の妥当性を判断する地理的（geographical）アプローチの 2 種類である．制約つきグラフ探索技術はグラフとしてのモデルを利用しているためグラフ的アプローチの旅行計画問合せにのみに適用することが可能である．

上に挙げた既存研究は，それぞれ 1 つまたは複数の POI を経由する旅行計画を求める問合せであるが，グラフを扱う寄り道探索は本研究を適用できるが，Oprical Sequential Route 探索と場所による軌跡探索は，地理的アプローチであり適用することはできない．本研究の次の適用検証には，複数 POI を経由するグラフ探索が重要な候補となりえる．

既存の旅行計画問合せに関する研究は，個別の問題を解くことに特化しており，単一情報源のメモリ・ファイル・データベースにすべての情報と能力が集中していることを前提としている．しかし，様々な情報やサービスを利用できるようになった現在では，既に提供されているサービスやデータベースを組み合わせることによって，新たな旅行計画サービスを迅速に提供することが求められる．本研究は，複数の情報源を組み合わせてより一般的な制約つきグラフ探索を実現するものであり，個別の問題によらず多くの応用に対して有効な手法であると言える．

2.2 データベース技術における位置づけ

2.2.1 異種分散データベース統合技術

データベースの実用化から今に至るまで，独立に構築されたデータベースを連携・統合して，当初の目的以外に利用したいという要求は常に存在してきた．例えば，販売管理のデータと顧客管理のデータを組み合わせて，売れ行きのトレンドを分析する等の例がある．このような新たな応用を行うためには，既存のデータベースシステムに改造を加えて実現することも可能であるが，その改造コストが大きいために，可能な限り既存システムに手を入れずにデータを連携するシステムに対する要求があった．そのようなデータベースは以下の 3 つの性質で特長づけされる．

分散性 サービスは一つのデータベースで構成されているとは限らない．鉄道網の情報と店の情報が別のデータベースで管理されている場合もある．

異種性 複数のサービスは独立に構築されているために，データ構造・データ項目の名前・データのフォーマット等は異なっている場合がある．

自律性 組み合わせる対象となるサービスやデータベースの自律性が尊重される．新サービスの都合で，対象側に変更が生じることは可能な限り避け

る必要がある．

この 3 つの性質を持つデータベース群（統合対象とするデータベースを要素データベースと呼ぶ）を仮想的に統合する技術は異種分散データベース統合技術と呼ばれる．本研究は，この異種分散データベース統合技術と深く関係しているため，以降の節でその関係を明らかにする．まず，統合対象の情報源は，時代による要求により発展してきているため，その観点での位置づけを示す．次に要素技術として重要な異なる情報源のスキーマ（情報の構造・形式・意味）の異種性を解消し統合するスキーマ統合技術との関係を示し，最後にデータ統合検索を実現するシステムの観点での関係について述べる．

2.2.2 統合対象の情報源

異種分散データベースにおける統合対象の情報源は，1980 年代までは圧倒的に関係データベースが主流であった．これらは大規模な計算機システム上で銀行業務や企業の販売管理システム等のビジネス用途に多く構築されていた．関係データベースの構造は表構造であり，その操作は SQL 言語によって行われる．SQL 言語は，関係代数という集合を基礎とした体系があり，その上に構築されている．また，関係データベース以前の技術としては，ネットワーク型（CODASYL）・階層型のデータベースが大規模なシステムには利用されていた．

1980 年代から 1990 年代にかけて，関係データベースの急速な普及とシステムのダウンサイジング化が進み，企業が持つデータベースの数とデータ量が急速に増えた．さらに，インターネットの普及により，Web の情報がデータベース的な情報源として利用され，データベース情報と Web 情報の連携の必要も増した．1990 年以降は，データベースの応用範囲もさらに広がり，標準化された情報の流通に利用する XML や，動画や静止画等のマルチメディア情報もデータベース化されるようになり，それらに対する研究が盛んになった．

まず，XML や Web 情報のような従来の表構造よりも複雑な木構造を表すデータベースについての研究が盛んとなった．また，これらのデータは構造が一様ではない（半構造）という点についても特長がある．XML を専門に高速に処理できる XML データベースが実用化された．XML データベースに対しての操作は，XQuery 言語が利用される．XQuery を高速に処理するためのデータ構造の構成法や，問合せ最適化の研究が行われた．2000 年以降は，静止画や動画をデータベース化するマルチメディアデータベースの研究が進んだ．これらでは，画像や動画を高速に検索する技術等が開発されている．市販データベース管理システムでは，これらをユーザ定義関数としてデータベース操作に組み込むように実装されることが多い．

さらに，応用分野の拡大に伴って，Spatial database[A1]，Temporal database[A2] という研究分野も確立した．これらは空間や時間を扱うデータベースを意味

し，応用としては，地図（空間情報），動画（空間＋時間），履歴（時間），ログ（時間）等があげられる．効率よく検索するためのインデックス技術や，問合せ言語．問合せ処理の最適化技術等の研究がなされている．

近年，交通網等を含む地図情報や，SNS におけるソーシャルネットワークや，Semantic Web の基盤として知識表現に使われる RDF グラフの管理等の要求から，この数年でグラフ情報を専門に扱えるグラフデータベース [A3] が急速に普及しつつある．Neo4j[A4]，AllegroGraph，InfiniteGraph 等の製品が世の中に出ている．

グラフデータベースに対する代表的な操作は，探索とパターンマッチである．探索の代表例は，幅優先・深さ優先探索や最短経路探索である．パターンマッチは，例えば，化学式の一致のような操作が相当する．RDF 検索言語である SPARQL は，RDF のトリプルのパターンをマッチさせ検索する言語であるが，グラフデータベースに対する検索言語の一例であると言える．

グラフデータベースではそのようなグラフ特有の操作を高速に行うためのインデックス技術やキャッシング技術が研究されている．旅行計画問合せに既存のデータベース技術を応用する試みは特に最近盛んに研究されている．最短経路探索を高速化するために空間インデックスを適用する研究 [A69] や，さらに複雑な旅行計画問合せとして前に述べた Optimal Sequenced Route Query [A66] もデータベースインデックス技術を活用している．

このように，連携したい情報源の種類は拡大しつつあり，その構造は徐々に複雑になり，情報の操作の仕方も，表構造に対する集合的な関係代数という単純なモデルから，XML のような木構造や，マルチメディア情報等の複雑な操作に進展してきている．しかし，グラフデータベースを統合対象とする研究は，その普及が始まったばかりであることから未着手である．本研究はグラフデータベースを対象とする異種分散データベース研究である点に新規性がある．

2.2.3 スキーマ統合技術

異種分散データベースでは，その統合検索を実現するために，様々な異種性を持つデータベースのスキーマを統合する必要がある．その技術は，スキーマ統合技術 [A6][A7] と呼ばれている．

スキーマ統合を実施するためには，情報源のデータモデルを，統合に必要な共通のデータモデルに変換し，一つに統一してから異種性を解消することが理論上無駄なく有利であると考えられる．そのデータモデルを共通データモデルと呼ぶ．

かつては，関係データベースが圧倒的に普及していたことと，データベース設計の主流な手法が ER モデルによっていたことにより，共通データモデルとしては関係モデルと ER モデルを利用する場合が多かった [A6]．しかし，

近年はスキーマ構造が木構造等の複雑な場合や知識処理的な論理記述を含む場合に拡張されている [A11][A13][A21]．その代表的な研究が，Madhavan 等によるスキーママッチングの研究 [A13] であり，その後の研究に大きな影響を与えている．本研究では，グラフデータベースのスキーマ統合を ER モデルに変換することで実現している．グラフデータベースを含むスキーマ統合は既存研究では行われておらずその点で本研究には新規性がある．

共通データモデルに変換した個々の情報源のスキーマはそれぞれ構造・命名・表現等の異種性を持っている．その異種性を解消し，ユーザが利用する統合スキーマを作成する技術をスキーマ統合と呼ぶ．本研究では，スキーマ統合という用語は，単に統合されたスキーマを作成するだけではなく，個々の情報源のスキーマと統合スキーマの対応関係を記述することも含めて広義の意味で用いることとする．

スキーマ統合の初期的な研究の動向については，Batini らのスキーマ統合に関するサーベイ論文 [A6] に詳しい．まずは，統合の前提となるスキーマ間の異種性の分類についての研究が進められた．ER モデル間の異種性の分類は [A6] で確立されており，関係モデル間の分類としては Kim らによる研究がある [A7]．

スキーマ統合の最終的な生成物は，異種性が解消されたすべてのスキーマをすべて含むようなスキーマとなる．それは統合スキーマと呼ばれる．しかし，その統合スキーマを作成することは，大企業等の巨大なスキーマや多くの異種性のある場合，かなり手間のかかる大変な作業であることが知られていた．そこで，その作業を整理するためのスキーマ統合の方法論の研究がなされた．また，その方法論を支援するツールの提案もなされた．しかし，統合スキーマ自体は，連邦データベースやメディエータにとって必須であるわけではない．Sheth らの連邦データベースのサーベイ論文 [A5] では，そのような統合スキーマを作成せずに，利用者が直接情報源のスキーマを検索言語を用いて利用する疎結合と呼ぶ連邦データベースのカテゴリが存在することを指摘している（それに対して統合スキーマを利用するものを密結合と呼ぶ）．このように，ある意味，統合スキーマの作成を断念してしまうような連邦データベースが検討されたのは，スキーマ統合が複雑なタスクであることが背景にある．このスキーマ統合の複雑さや稼働を軽減することは大きな課題となっていた．

しかし，スキーマ統合の稼働を軽減するような研究は，生産性向上という意味でややソフトウェア工学的な観点であり，正しい結果を求めたり，如何に結果を早く求めるかということが中心であるデータベース工学の分野ではあまり進展していなかった．

本研究では，概念グラフを利用したスキーマ統合技術により，スキーマ統合のツール化を利用した方法論を確立しており，スキーマ統合稼働の軽減に貢献している点で価値がある．

2.2.4　データ統合技術

異種分散データベース技術を利用して仮想的なデータ統合検索を行うシステム技術としては，1990 年前後まで，主に関係データベース等のレガシーな情報源を連携させる連邦データベース（federated database）技術が発展した．独立に構築された自律的で異種性を持つ分散したデータベースを連携する技術として定義される．(連携対象となるデータベースを要素データベースと呼ぶ)．Sheth らの連邦データベースに関するサーベイ論文 [A5] によって，その基本的な考え方が示されている．連邦データベース技術では，問合せ実行の最適化技術が重要な課題であった．可能な限り要素データベースに処理をまかせるほうが，通信コスト上も計算処理リソース上も有利なケースが多かったために，例えば，検索条件を要素データベースに処理させ絞り込んでから連邦データベース側で処理する等の戦略の最適化が重要であったからである．

次に，1990 年代以降進化した複雑な構造やマルチメディア等のより多様な情報源も連携対象とする連邦データベースよりも進んだ情報源の統合は，メディエータ技術と呼ばれることがある [A8][A9][A19][A12]．その代表例としては，TSIMMIS プロジェクト [A8]，Information Manifold プロジェクト [A9] がある．メディエータの対象とする情報源は，従来の RDB に加えて，XML・Web ・マルチメディア情報等があげられる．メディエータの特長は，木構造・半構造等といった構造の複雑性の違いもあるが，Web 情報やマルチメディアデータベース等にある情報源の能力のばらつきを考慮する技術が特に重要となっている．Web 情報では必ずしもすべての項目に条件を指定できなかったり，画像検索ではシステム毎に指定できるパラメータが違うということが例としてあげられる．情報源の能力のばらつきがあると，問合せの最適化の戦略に変化が出てくる．連邦データベースでは可能なかぎり要素側で実行させるという説明を述べたが，メディエータでは，要素情報源の能力を考慮したきめ細かい制御によって，問合せの最適化を行う必要が出てくるからである [B78]．

近年の異種データベース統合技術の研究については，2010 年の ETH Group によるレポート [A10] にその最新動向がまとめられており，特に論理記述や知識処理を用いた研究 [A11] が盛んになっていることが特長としてあげられる．また，ラッパーの構成法 [A12] 等の研究もあるが，近年応用が進展しているグラフ情報を対象情報源としたり，グラフ情報に特有な探索操作やパターンマッチ操作を組み合わせて情報統合をしたり，問合せ最適化を行ったりする研究までは，至っていないというのが現状であると考えられる．また，関係データベースにおける SQL 処理，XML データベースにおける XQuery 処理のように，データベース側に汎用的なデータベース問合せ言語が存在することを前提としていた．

本研究は，既存のメディエータ技術では実現していなかったグラフデータ

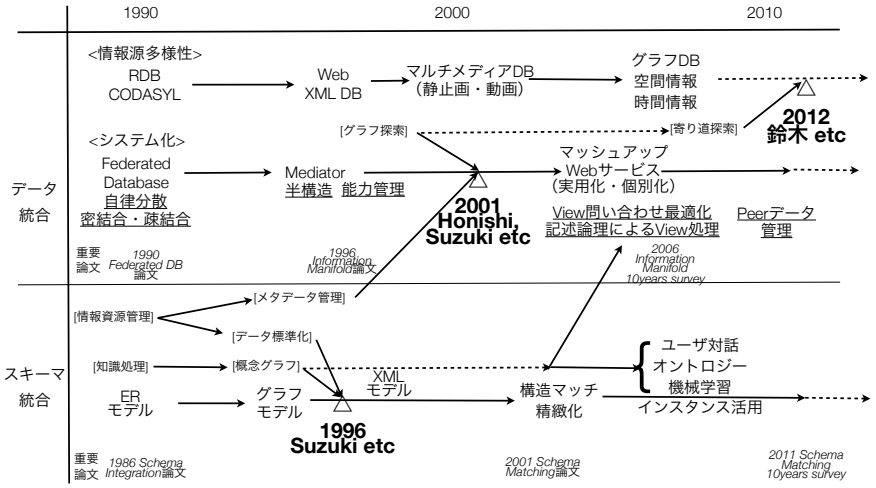


図 2.1: 既存研究における本研究の主著の位置づけ

ベースを対象とした情報源の能力のばらつきを考慮した最適化を行えることに特に新規性がある．データベース側に汎用的な問合せ言語は想定しておらず，グラフ探索機能が仮想的な表として得られるという仮定に基づいており，問合せ言語が確立していないグラフデータベース一般に対して適用することができる．また，動的なデータ統合検索を適用していることも特長であり，動的に生成される検索候補を選択する本方式は，末端ユーザによるスキーマ統合への介入の一種である．スキーマ統合における個人適応やユーザアクションの利用は，現代における情報源統合の重要課題の一つであり (2006 年の Alon Halevy のサーベイ [A19])，その可能性の一端を示している．

本研究は，旅行計画作成等のグラフ技術の活用，情報源の多様性の進展の流れと，メタデータ管理方式の進化等の関連研究と結びつけられている．各技術の主著の関連研究との位置づけを図 2.1 に示す．

第3章　探索対象情報源のスキーマ統合技術

3.1　導入

本章では，動的なデータ統合検索を行うために利用する統合されたスキーマの概念グラフ等のメタデータを得るためのスキーマ統合技術に関して述べる．スキーマ統合は，企業全体で管理しているデータを把握し企業全体での鳥瞰を得たり，情報資源の有効活用のために既存データベース（以後 DB）を再編，データ流通するために重要とされ研究されてきた．既存 DB のスキーマ統合は Batini ら [A22] によって次のようなステップに整理されている．

- ステップ 0：統合対象スキーマの同じデータモデル（共通データモデル）への変換．
- ステップ 1：スキーマ間で対応するスキーマ要素の発見．
- ステップ 2：対応するスキーマ要素間の不一致・衝突（conflict）の発見．
- ステップ 3：不一致・衝突の解消
- ステップ 4：調整したスキーマのマージ．

従来のスキーマ統合では，スキーマ間で不位置．衝突が存在する場合の対処法であるステップ 2，3 が研究されていただけで，スキーマ間で対応するスキーマ要素を探し出すステップ 1 は，統合スキーマ設計者の総合的判断によって行われていた．スキーマの規模が大きくなると，かかる手間は膨大であった．

類似スキーマ要素の発見には，スキーマ要素間の類似性を何らかの尺度で表現し，計算機に酔って処理して統合スキーマ設計者の負荷を軽減することが重要である．

本章では，類似スキーマ要素を発見することを中心とした支援を計算機で行うための方法を提案し，それに基づいたスキーマ統合支援ツールについて述べ，このツールを用いた場合のスキーマ統合手順を提案する．以下，2 節で，スキーマ要素発見の問題を整理し，それを本章の提案でどう解決するのかの概略を述べる．3 節から 5 節は，その解決のための各手法をスキーマ要素名標準化，概念グラフへの変換，スキーマ要素間類似度計算の順に述べる．

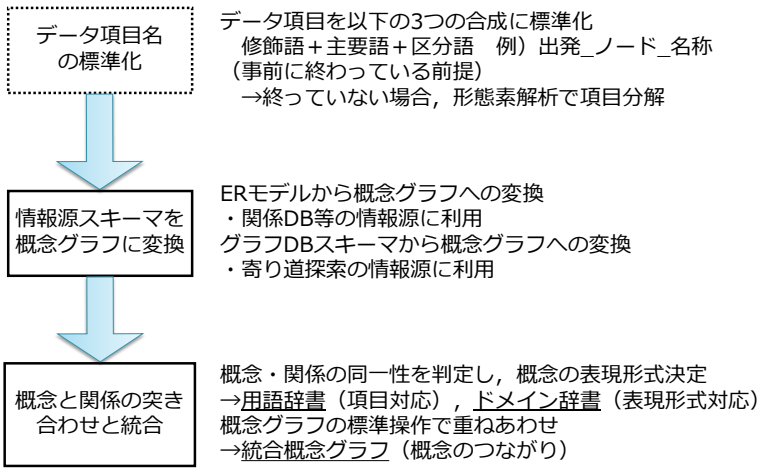


図 3.1: スキーマ統合の概要の手順

グラフデータベースからの概念グラフの変換については，4 節の中で述べる．6 節ではスキーマ統合支援ツールと，このツールを用いた場合のスキーマ統合手順を説明する．7 節で関連研究との関係を考察する．

スキーマ統合の概要手順を図 3.1 に示す．このスキーマ統合の過程で得られるデータ項目等の対応関係のメタデータを次章以降のデータ統合検索で利用する．本スキーマ統合技術によるアウトプットを図 3.2 に示す．

3.2　類似スキーマ要素発見の問題と解決案

スキーマ要素間の類似性は以下の 3 種類に分類される [A23].

名称の類似性　スキーマ要素の名前が似ていること．例えば，共通する文字列を多く含むスキーマ要素は類似していると解釈される．

意味の類似性　スキーマ要素に数量化された意味が定義されており，その意味が類似していること．

構造の類似性　スキーマ要素の周辺に似た概念があれば，そのスキーマ要素は類似していると考える．例えば，共通した属性を多く含む実体型は，類似しているとみなす．

これらの各類似性に関して類似スキーマ要素の発見という点で問題がある．このうち，本論文が取り組むのは名称類似性と構造類似性に関する問題である．意味類似性についての考察は 7 節で行う．

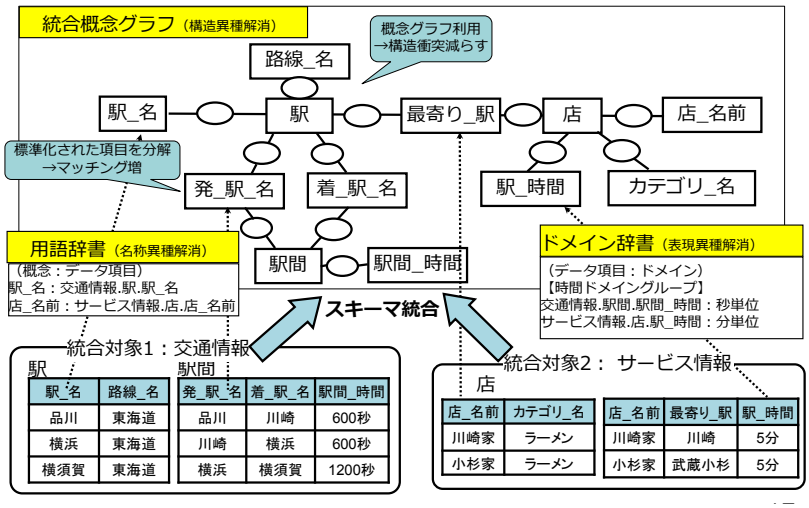


図 3.2: スキーマ統合の概要とアウトプット

3.3 スキーマ要素名の標準化

本技術の第一のポイントは，その前段でスキーマ要素（実体型・関連型・役割・属性）の名称を標準化する点にある．属性名はデータ項目名の標準化手法として知られる命名規則 [A28] によって整備し，実体型名，関連型名，役割名にはその規則を変更したものを用いる．本章では，この規則に基づいてスキーマ要素名を標準化する手法について述べる．

3.3.1 Durell-関根のデータ項目命名規則

データ項目とは，計算機に格納するデータの最小単位である．そのデータ項目の標準化は，企業全体で，情報システム間で共用されるデータベースやファイルのデータ項目が次の要件を満たすようにすることと言われている [A28].

要件 1 定義域（データ項目がとり得る値の範囲）が同じデータ項目は，表現形式（型，けた数，コード値）も同じである．

要件 2 定義域が同じで内容も同じデータ項目は，日本語名も同じである．逆に，内容が異なるデータ項目は日本語名が異なる（識別性の保証）．

要件 3 日本語名は誰にもわかりやすい，自然な日本語である（認識性の保証）．

データ項目名（例えば,「新規お客様番号」,「顧客名」,「事業所住所」など）の標準化は，データ項目が上記要件を満たすようにするために中心的な役割を果たす．Durell-関根のデータ項目命名規則 [A28] は，Durell が上に示した

要件 3 を英語について具体化した命名規則 [A29] を，関根らが拡張し，規則に合った名称であるか否かが容易にチェックできて，しかも，日本語による自然な命名ができるようにしたものである．規則はおよそ次のようである．

規則 **1** データ項目名は,「コード」,「名称」「番号」等のデータ項目の目的を示す用語（区分語 C）を一つもつ．

規則 **2** データ項目名は，それが組織内のどの実体にかかわるものであるかを示す用語（主要語 P）を一つもつ．

規則 **3** データ項目名は，意味を補うための用語（修飾語 M）を複数もつことができる．

規則 **4** 用語の並びを規定する．日本語の特徴を考慮し，修飾語＋主要語＋区分語という語順とする．

規則 **5** 区分語，主要語，修飾語のそれぞれには，あらあじめその用途で規定している用語のみ用いる．(各用語がどの用途に使えるかを規定したものを用語種別と言う.)

例えば，主要語だけで区分語のない「電話」では，電話機の種別なのか電話番号なのか，またはそれ以外なのかを判断できない．これらの規則はわかりやすい名前を付ける上で必須の条件である．規則 4，5 によって命名者以外でもどの用語が区分語，主要語，修飾語なのかの判別ができる．以後便宜上，用語の区切りにアンダースコアを入れることにする．

3.3.2 用語辞書を用いたデータ項目名標準化手法

関根らは，上記の規則を考案すると共に，この規則によってデータ項目名標準化を行うために用語辞書を用いた標準化手法を開発した [A28]．それによれば，データ項目名を構成する各用語について，それらの表現上の差異（例えば「ユーザー」と「ユーザ」と「USER」）を吸収するために，標準用語（例えば「ユーザ」）を決める．更に，類似の標準用語（例えば「ユーザ」「顧客」「加入者」）をグループ化して，そのグループを代表する標準用語（例えば「顧客」）を決める．用語辞書には用語と標準用語の対応表，標準用語の用語種別，標準用語と代表標準用語の対応表を格納する．

このような用語辞書を作成・管理し，活用することで，任意に与えられたデータ項目名がこの規則に合っているか否かをチェック（データ項目名のチェック処理）できる．また，同一の定義域，あるいは，同じ内容をもつデータ項目を見つける（類似データ項目分類処理）ためのアルゴリズムを考案し，この規則に従って命名したデータ項目名が要件 1 と要件 2 を満たしているかを容易にチェックできる．

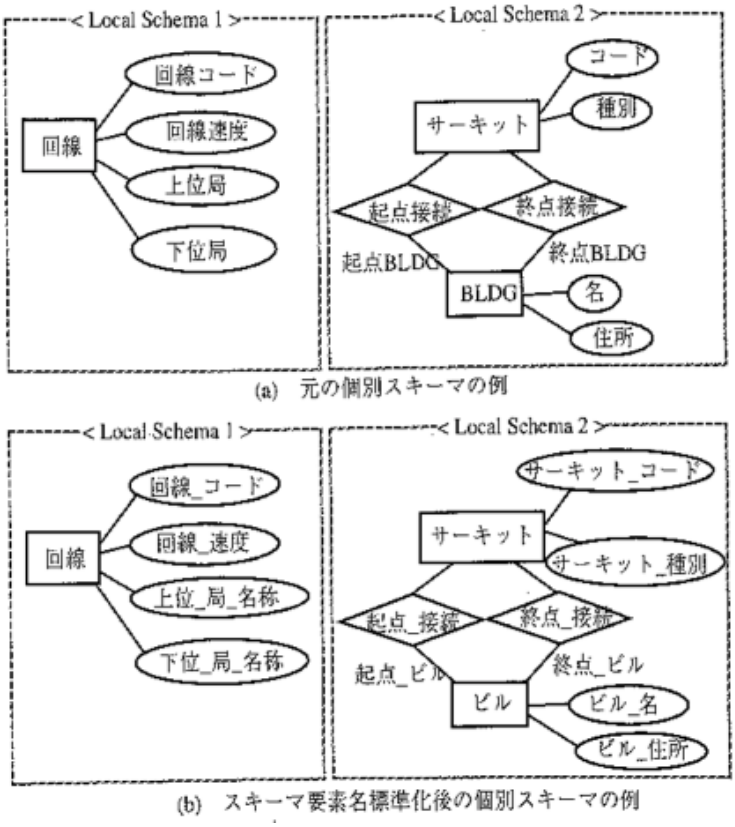


図 3.3: 個別スキーマ表現の例（ER モデル）

3.3.3 スキーマ統合におけるスキーマ要素名標準化

本章の手法では，名称の構文を規定しその構成要素に用いる用語を用語辞書で整理する上記の関根らの手法に準じた方法によりスキーマ要素名を標準化する．具体的には，属性名については上記命名規則と照らし合わせて，不足している区分語や主要語を補う．このとき，用語辞書にある標準用語を使用するようにする．実体型，関連型，役割の名称は，(修飾語＋）主要語という形になるよう，属性名と同様に修正を行う．図 3.3 に統合対象のスキーマの例と，それらをスキーマ要素名標準化した結果を示す．このようなスキーマ要素名の標準化をスキーマ統合に先立ち行うことがポイントである．

NTT 社内の通信網管理関係の 11 の DB について約 11000 データ項目を分析した結果，使われている用語は延べ約 3700 用語，11 システム全体での用語種類は 700 余で．更にこれから標準用語にしぼり込むと種類は 200 余りなることがわかった．従って，用語辞書を順次整備し，その用語を用いてスキー

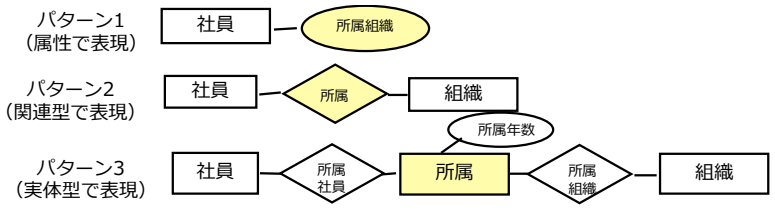


図 3.4: 構造異種の例

マ要素名を修正することにより，類似スキーマ要素の発見が容易に行えるようになる．2 節で述べた従来の方法のように，スキーマ要素の名称そのもののマッチングや類似辞書の援用で比較するものに比べて，本手法は，名称の構文を決めることで，用語が限定でき，それらの類似性の整理だけでスキーマ要素間の名称類似性の判定が可能になるという利点がある．なお，関根らのアプローチと同じく，用語辞書にない用語は標準化を進めながら順次登録していくこととし，用語辞書の完備を必ずしも前提としていない．

3.4 概念グラフへの変換

3.4.1 スキーマ統合で解消すべき異種性と構造異種

スキーマ間の異種性は以下のように整理されている [A6].

名称異種 同じ「概念」を異なる「名前」で表すこと．

表現異種 同じ「概念」を異なる「データ表現形式」で表すこと．

構造異種 同じ「概念」を異なる「型（データモデル構成要素）」で表すこと．

ER モデルにおける構造異種の例を図 3.4 に示す.「所属」という概念をそれぞれ属性・関連型・実体型と異なるデータモデル構成要素で表現している．

3.4.2 ER モデルからの概念グラフ変換

構造衝突（構造異種）の発生しにくいモデルを用いれば，構造の類似性も単純な形で考えることができるようになる．構造衝突を発生しにくくし，スキーマ要素の構造的類似性を簡潔にするには，

- 構成要素が最小限
- ドメインと従属性を明確に分けている

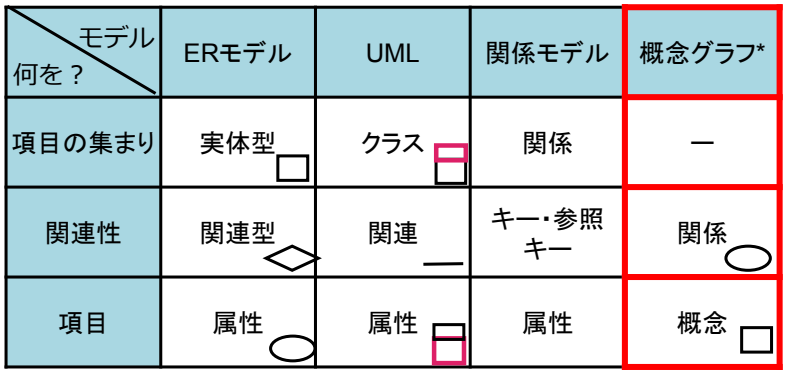


図 3.5: データモデルとその構成要素

という条件を満たすモデルを用いることが望ましい．本章ではこの条件を満たすモデルとして，概念グラフ [A30] を用いることを提案する．概念グラフは，実世界を概念にあたる概念（concept．図での表記は長方形）と，概念間の関係を表す関係（relation．図での表記はだ円）のみによってモデル化するものである．関係には方向があり矢印で表記する．データモデルの構成要素についての比較を図 3.5 に示す．概念グラフでは,「項目の集まり」に相当する構成要素を持っていない．よって，概念グラフに変換する過程で構造衝突の現象が期待できる．

本章では，3 節の標準化によって実体型名，関連型名，役割名，属性名を整理した ER モデルの形の個別スキーマを概念グラフに変換する方法を説明する．提案の変換法では，スキーマ要素名標準化において識別した区分語，主要語，修飾語を利用して概念の取出しを行うことがポイントである（7 節で考察する）．

実体型，関連型，役割の場合 実体型は同じ名前をもつ概念に変換する．関連型は，属性を持つ場合については単独の概念を生成し，関連型の役割の方向に従って関係に変換する．

属性の場合 基本的な考え方は次のようである．

- 主要語は概念の候補となる．修飾語が付いている場合は，主要語はそれ単独で概念として生成する．しかし，属性名の主要語はその属性をもつ実体型の名前と等しい場合が多く，その場合，主要語の表す概念の生成は実体型からの生成と重複することになるので，概念は生成しない．
- 修飾語は，主要語とその属性をもつ実体型との関係を表すと解釈することができるので，概念間の関係に変換する．修飾語はオプ

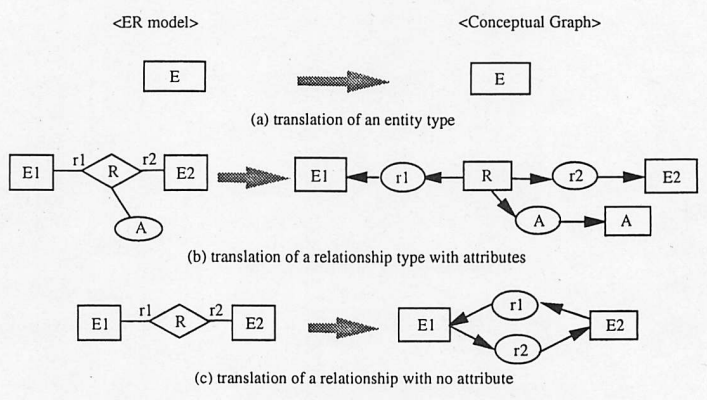


図 3.6: ER モデルから概念グラフへの変換規則（1/2）

ションであるために，ない場合もある．修飾語がない場合は，実体型名と主要語が等しくなければ，主要語を関係名とする．

- 区分語は単独では一つの概念を表すとは考えられないので，独立した概念にはしない．

以上の考え方に従って作られて変換規則を図 3.6，3.7 に示す．この変換法を適用して，図 3.3 の例を概念グラフに変換した結果を図 3.8 に示す．

3.4.3 グラフデータベースからの概念グラフ変換

グラフデータベースのデータモデルは，一般的に以下の構成要素を持つ（グラフデータベースの管理システムによっては，その呼び方が異なることがあるが対応をとることができる.）.

- ノード：交通路における駅等を表す
- エッジ：交通路における駅間の接続情報等を表す
- プロパティ：ノードとエッジの持つ性質を表す

ノード（エッジ）の持つプロパティは，すべてにおいて共通である必要はない，そのようにデータ毎に持つ性質（属性）が異なるデータモデルは，スキーマレスであると呼ばれる．例えば，XML はスキーマを定義することもできるが，定義せずスキーマレスとして利用することもできる．

本研究では，グラフデータベースと既存データベースの統合を実現するために，グラフデータベースのスキーマを作成するアプローチをとる．データ統合検索を行うために，スキーマレベルの対応づけを行う．ユーザからの検

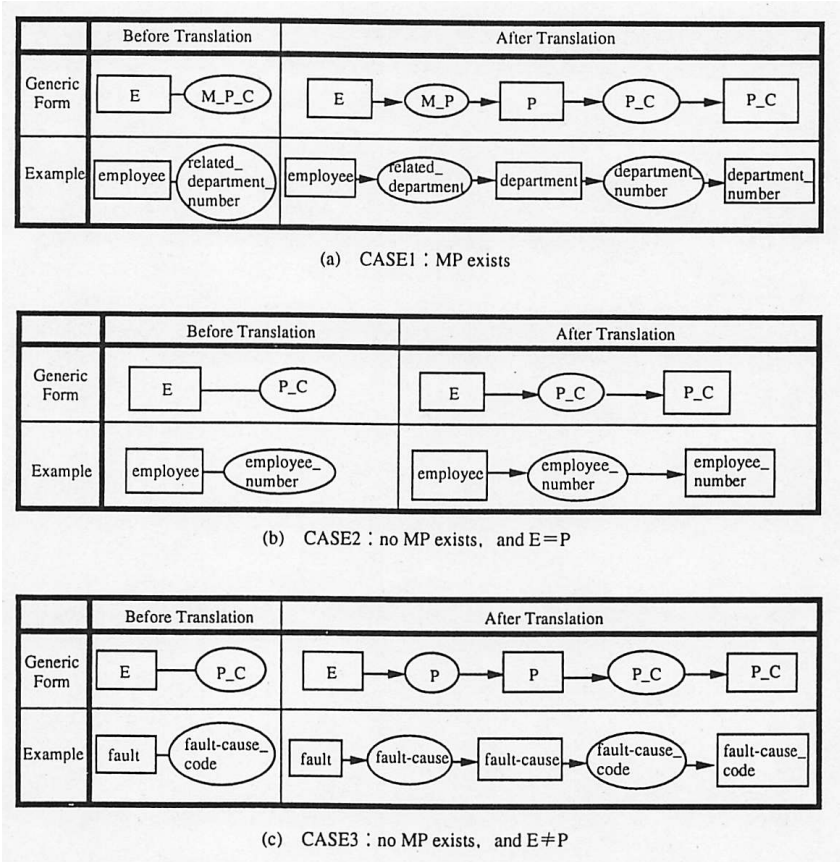


図 3.7: ER モデルから概念グラフへの変換規則（2/2）

索要求に対してデータ項目が存在するグラフデータベースを見つけ，検索を実行する．

グラフデータベースのスキーマ作成の基本方針は,「ノード」「エッジ」「機能」に相当する実体型を持つ ER モデルを作成し，前節で述べた ER モデルからの概念グラフ変換手法を適用することである．その手順を図 3.9 に示す．

まず，ノードとエッジに相当する実体型を作成し，ノードとエッジが持つプロパティを属性として追加する．これはグラフをトラバースする等の手段によって作成する（ノード・エッジにより持つプロパティが異なる可能性があるため）．また，駅ノードと店ノードのように種類の異なるノードは区別する．

次に機能実体型を作成する．これは，例えば，ダイクストラ探索や寄り道探索のようなグラフデータベース機能を仮想的な表としてとらえた実体型である．機能のパラメータと機能の出力を属性として生成する．

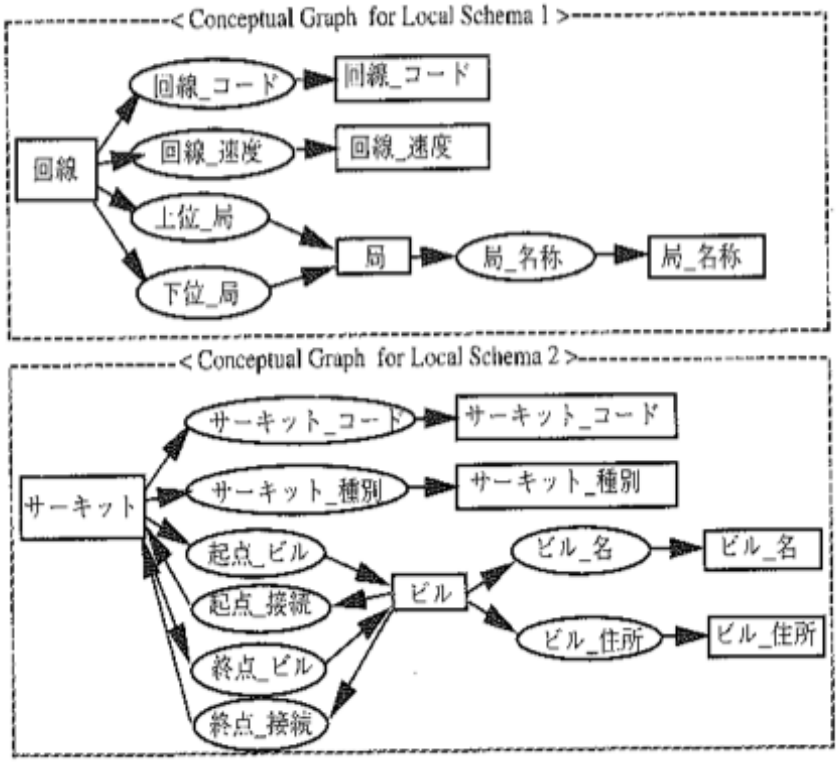


図 3.8: 概念グラフへの変換結果例

以上で ER モデルが作成されるので，前節に述べた手法によりそれを概念グラフに変換する．

3.5 スキーマ要素間の類似度計算法

概念グラフを構成する概念については，単独での類似度（本章では名称の類似度に限定する）と，周辺との関係による類似度を分けて，次の 3 段階で計算する．

ステップ **1** 名称類似度を求める．このとき，名称の整備に用いた用語辞書にある用語についてのグルーピング情報を利用することができる．

ステップ **2** 名称による単独での類似度の結果を利用して，周辺関係による類似度を計算する．

ステップ **3** ステップ 1 の名称類似度とステップ 2 で求めた周辺関係の類似度との加重平均を総合的類似度とする．

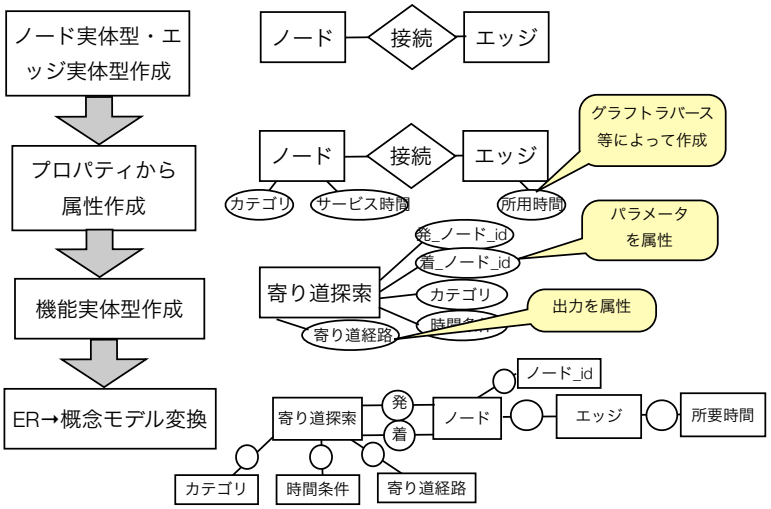


図 3.9: グラフ DB スキーマ作成と概念グラフへの変換

関係については，概念の同一性が確定したのち，二つの同一の概念の間にある関係を類似候補として吟味することで，周辺類似度を考慮し，次に関係についての名称類似度を求めて総合的類似度を決定する．

以下，概念の類似度の計算法を説明する．

3.5.1 名称類似度の計算

図 4-2 によると， 概念名のパターンとしては,「E」「P」「P_C」の三つの場合がある．よって，概念名の名称類似度を求める場合,

Case 1 概念名が両方とも分解していない場合（つまり,「E」対「E」,「P」対「P」,「E」対「P」の場合）

Case 2 概念名が両方とも分解している場合（つまり,「P_C」対「P_C」の場合）

Case 3 概念名の一方が分解してなく，もう一方が分解している場合（つまり,「P」対「P_C」,「P_C」対「P」の場合）

の 3 通りがある．ここで，Case 3 は，概念グラフへの変換法により「P_C」と常にペアで必ず,「P」という概念が生成されることから（図 4-2(e) の場合は，P が存在しないように見えるが，P=E であることに注意）,「P」対「P_C」,「E」対「P_C」は類似しているとみなす必要はなく，この組合せに対しては，類似度を 0 とする．以下に Case 1 と Case 2 について名称類似度の計算法を述べる．

Case 1：概念名が分解していない場合 同じグループの標準用語になっているか否かで類似度を決める．比較する二つの概念名を入力し，それらに対応する代表標準用語を検索する．二つの概念名の両方に対応する代表標準用語があって，かつそれらが等しければ，名称類似度は 1（最大値）に設定される．それらが等しくなければ，それらは全く別ものであるので 0（最小値）に設定される．一方しか代表標準用語が存在しない場合やどちらにも代表標準用語が存在しない場合は，部分列マッチングなどの手法を用いて類似度を求める．このアルゴリズムを先の例に適用すると，概念名「局」と「ビル」は代表標準用語として両方「ビル」が見つかるとすると類似度が 1 となる.「サーキット」は代表標準用語が用語辞書に登録されていないとする．すると「回線」と「サーキット」は代表標準用語が片方しか見つからず，部分列マッチングも起こらないので類似度は 0 となる．

Case 2：概念名が分解している場合 区分語，主要語のそれぞれについて対応する代表標準用語が異一するかどうかを判定し，両方とも一致するものには，類似度として 1 が設定される．主要語のみが一致する場合は，区分語のみが一致する場合の類似度 N2 に比べて高い類似度 N1 が設定される．両方一致しない場合は最も低い類似度 N3 が設定される．対応する代表標準用語がない場合の類似度は，部分列マッチングなどの手法を用いて計算される．このアルゴリズムを先ほどの例に適用すると，概念名「回線_コード」と「サーキット_コード」は，主要語の分類が等しくなく，区分語が等しいので，中間の類似度 N2（$0 < N2 < 1$）が設定される．

3.5.2 周辺類似度の計算

次にこのように計算された名称類似度を用いて，周辺の類似度を計算する．概念 A，B に対して，それぞれ隣接している概念の集合を SA，SB とする．周辺の類似度は，以下の式によって計算される．

$$(A \text{ と } B \text{ の周辺の類似度}) = \frac{sum(max(SA, SB \text{ 内概念間の名称類似度}))}{min(SA \text{ の個数}, SB \text{ の個数})}$$

ここで，$(SA \text{ の個数}) \leq (SB \text{ の個数})$ で，max は SB における最大，sum は SA についての総和（逆の場合も同様）.

図 4-3 の例の「回線」（= A とする）と「サーキット」（= B とする）の場合を考える．この二つは名称の類似度が 0 であった．これらの隣接する概念はそれぞれ

$$SA = \{ \text{回線\_コード}, \text{回線\_速度}, \text{局} \}$$

$$SB = \{ \text{サーキット\_コード}, \text{サーキット\_種別}, \text{ビル} \}$$

である．SB 内の概念と SA 内の概念の名称類似度を求めると,「サーキット_コード」に対しては,「回線_コード」が最大になる（類似度は N2 となる）．「サーキット_種別」に対しては,「回線_速度」が最大になる（値は N3 である）．「ビル」に対しては,「局」が最大になる（値は 1 である）．よって，N2=0.30, N3=0 とすると「回線」と「サーキット」の周辺類似度は約 0.43 となる．

3.5.3　総合的類似度の計算

最終的には，名称類似度と周辺類似度の加重平均を類似度とする [A23]．上記の「回線」と「サーキット」の例の場合，名称の類似度が 0 であり，周辺の類似度が 0.43 であるから，例えば重みを（名称）:（周辺）を 1:1 にすると最終的な類似度は，0.30 となる．名称類似度のみでは類似とみなされなかった概念が，周辺の類似度を考慮することにより類似度が高くなり，設計者が類似とみなす助けとなる．重みは概念名のパターンに応じて変えることも考えられる．例えば，Case1 では周辺類似度の方に重みを増やすなどである．

3.6　スキーマ統合の手順と支援ツールの実現

3.6.1　スキーマ統合支援ツールを用いた統合手順

以上の方法に基づいて，スキーマ統合支援ツールを作成し，このツールによる支援を前提としたスキーマ統合手順を確立した．本章では，スキーマ統合支援ツールを用いたスキーマ統合の手順と，本手順の各作業におけるツールの役割を述べる．図 3.10 にツールの構成を示す．ここでユーザインタフェース部は，すべての機能を呼び出すためのメニュー，類似性計算結果の表示，概念グラフの概念や関係の同等性の判定結果の入力，スキーマの表示などの機能をもつ．

手順 **1**：スキーマ情報読み込み　スキーマ統合支援ツールの入力は，ER モデルで表された統合対象のスキーマ情報である．ER モデルを形式的に表現する言語を用いてツールに取込む（スキーマ情報取込み部）．グラフデータベースのスキーマは事前に，データのトラバースを利用して先に述べたルールに従い ER モデルとして生成されているとする．

手順 **2**：スキーマ要素名標準化　標準化部が 3 節に述べたスキーマ要素名標準化を設計者が行うことを支援する．基本的にスキーマ要素名標準化のもととなったデータ項目名標準化の計算機による支援法 [A28] を利用する．3.2 節で述べた用語辞書を用い，名称チェック処理を行う．属性名の場合，区分語が欠けている場合は，その属性のデータ型や値の例から区分語を推定し付加する．主要語・区分語両方とも欠けている場合は，

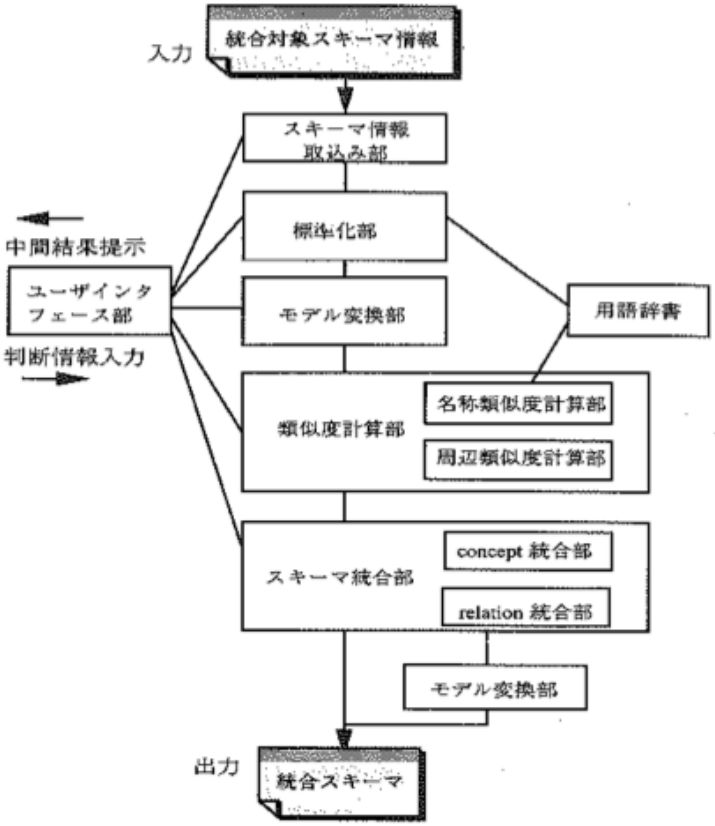


図 3.10: スキーマ統合支援ツールの構成

それ全体を主要語とみなし，区分語が欠けている場合と同様の方法で，区分語の付加を行う．また用語辞書にない用語は，登録処理をその都度行い，後から行う統合作業に役立てる．実体型名，関連名，役割名についても同様に行う．

手順 **3**：概念グラフへのモデル変換 名称が標準化された個別スキーマは，モデル変換部によって 4 節の変換ルールに従い自動的に概念グラフに変換される．

手順 **4**：概念と関係の突き合わせと同等性の確定 変換された複数の概念グラフの概念と関係を突き合わせ同等性を確定する．その手順を図 3.11 に示す．

まず，5 節で述べた計算法によって概念の類似度を計算する（類似度計算部）．名称類似度と周辺類似度から総合的類似度を計算する際の両者

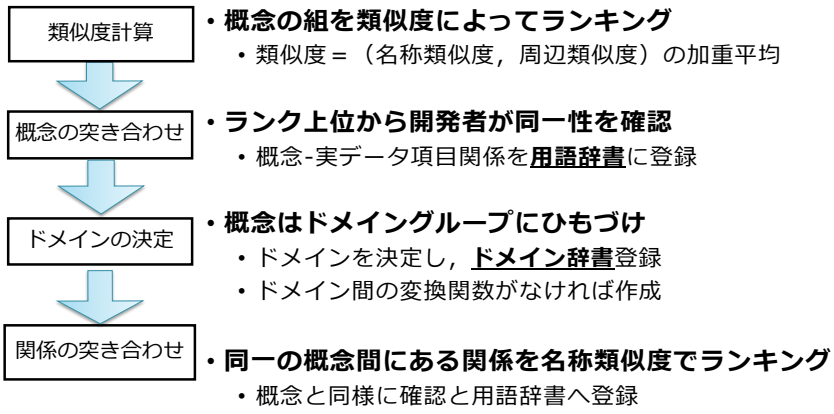


図 3.11: 概念と関係の突き合わせと統合手順

の重みは，概念名のパターンや用語辞書の整備状況などを考慮できるように，設計者が任意に与えられるようになっている．

次に類似度計算結果を参考に，概念の突き合わせを行う．類似度計算結果は，類似度の高い順にソートされて統合スキーマ設計者に提示される．その結果を見て，設計者は概念の一致性，包含性を判断し，指定する．設計者に指示された同等性をもとに必要に応じて再度類似度計算を行うこともできる．どの概念名を統合スキーマに採用するかも設計者が指定する．その結果により個別スキーマのデータ項目が統合スキーマ（統合概念グラフ）のどの概念に対応するかの情報が得られるため，それをメタデータの一部である用語辞書に記録しておく．

次に概念とそれに対応するデータ項目のドメインの決定を行う．ドメインとは抽象的な表現形式を表し，同じ意味を表す表現形式の集まりであるドメイングループによりグループ化されている（詳細は文献 [B82]）．ドメインの例を図 3.12 に示す．ここではレビューサイトにおける店の評価点の表現を，数値・英字・☆の数で表す場合の例であり，数値等がドメインであり，店評価点全体がドメイングループとなっている．矢印は変換関数の存在を示している．概念をドメイングループに対応させ，対応する情報源のデータ項目のドメインを決定する．ドメイン間の変換関数がなければ作成する．この関係はドメイン辞書に登録する．

最後にツールが概念の突き合わせ結果を反映して，関係の類似度を再計算しランキングして設計者に提示する．設計者は，その結果を利用して比較し，一致性，包含性を判断して，対応を用語辞書に記録する．

手順 **5**：概念グラフの統合 手順 4 で指定された結果をもとに，概念グラフ

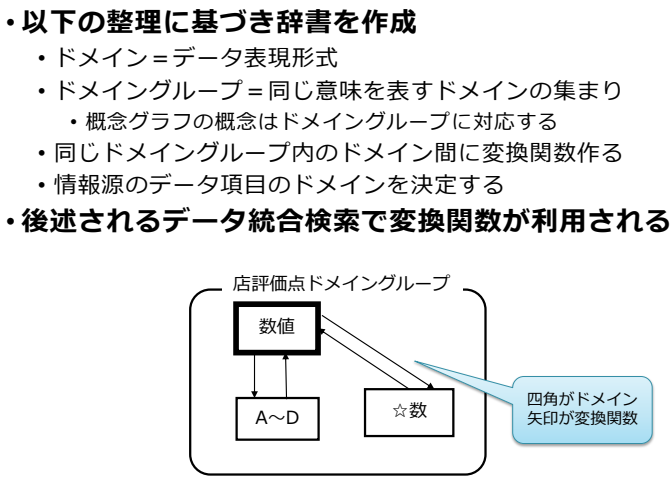


図 3.12: ドメイン辞書

をマージする．このマージには二つの概念グラフの操作である join と simplify がそのまま適用できる．join は，同一と判定された概念を同一視する．次に simplify により，join された結果生じた関係の冗長を排除する（アルゴリズムの詳細については，文献 [A30] 参照）．

手順 **6**：**ER** モデルへの変換 必要であれば概念グラフの形で得られた統合スキーマをモデル変換部で ER モデルの形などに変換する．正しく，ER モデルへ変換するためには，ER モデルの持つ制約を守る必要がある．例えば，関連型と関連型が直接結びつくことはない等である．本方式において概念グラフへの変換では，概念を分割したり等が行われているために，ER モデルへ戻す変換は自明ではない．そこで，概念グラフ上の概念と関係が元々どの ER モデル上の構成要素で表現されていたかという由来を考慮し，かつ実体型と属性両方が由来であれば実体型を優先する等のルールを設けて正しく ER モデルへ変換できるアルゴリズムを開発している（詳細は [B80] 参照）．

図 3.13 に図 3.8 の概念グラフを統合した結果を示す．

3.6.2 スキーマ統合支援ツールの効果

このスキーマ統合支援ツールを用いた，稼働の削減効果を見積もった．まず，ツールを用いて，いくつかの例題を解いたところ，類似スキーマ要素の発見が支援され，個々の類似判定・調整は 3 分程度しかかからないことがわかった．4.3.3 で述べた通信網管理関係の 11DB の例では，本ツールの支援な

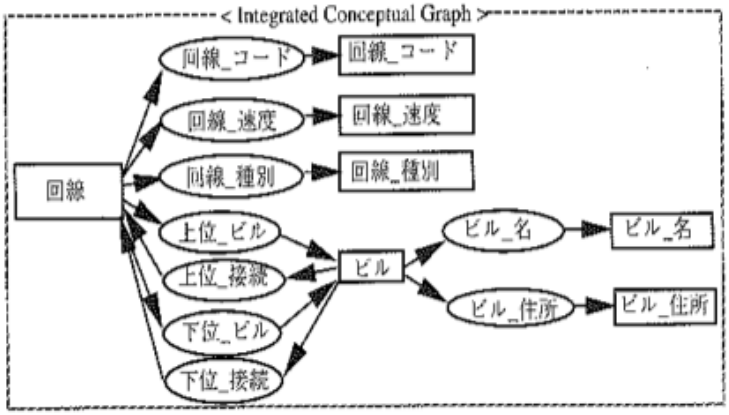


図 3.13: スキーマ統合結果例

しで統合スキーマ（70 実体，500 属性）を作成するのに要した稼働は 860 人日（内訳：データ項目の整理に 240 人日，データ関係分析に 340 人日，統合化段階で統合実体とそれらの属性の決定整理に 280 人日）であった．これを本支援ツールによって支援した場合の稼働を推定する．最初の項目整理の稼働は同じである．標準化の稼働は 700 用語の整理にかかる稼働として 20 人日となる．また類似判定・調整については，データ項目数にほぼ等しい回数の類似判定・調整があるとし，類似度計算やマージ計算のマシンタイムは無視すれば 60 人日かかる．合計稼働は 320 人日となる．再計算・判定（60 人日×繰り返し数）を多少繰り返しても従来の稼働に対してかなりの削減が見込める．

3.7 関連研究との比較

3.7.1 概念グラフを用いたスキーマ統合の従来手法

概念グラフを共通データモデルとして用いる方法は，Creasy ら [A31] によって検討されている．そこでは ER モデルから概念グラフに変換する方法の概略が述べられているが，大きく分けて二つの問題があった．

問題 1：関連型の変換の問題 Creasy らの方法では，一つの関連型は一つの関係に変換される．しかし，関連型の役割を何に写像するかは不明である．また，概念グラフの関係は方向性をもっているが，関連型を関係に変換するときに，方向がどうなるのかは明確でない．本章の方法では，これらを明確にした．

問題 2：**属性の変換の問題** Creasy らの方法では，一つの属性を，その属性と同じ名前をもつ関係と概念の組合せに写像している．しかしこの方法では，適当な概念を生成できないことがある．例えば,「従業員」という実体型が「所属組織」と「関連組織」という属性をもつとする．上記の方法で変換すると,「所属組織」と「関連組織」という二つの異なった概念が生じる（変換 1）．しかし，その二つの概念は本来「組織」という一つの概念であり,「所属」と「関連」は概念「従業員」と概念「組織」の関係と解釈するほうが自然である（変換 2）．実際，これを「組織」が単独にモデル化されたスキーマと統合しようとすると,「所属組織」と「組織」は名称矛盾になってしまう．このような状況が生じる原因は，先に述べたとおり ER モデルにおける属性が,「現実世界を表す概念」と「概念間の従属関係」の二つが混在したものになっていることにある．4 節で述べた変換法では ER モデルを概念グラフに変換するときに，その混在を分離することがポイントである．

3.7.2 意味の類似性

意味の類似性については，例えば，北川ら [A32] は，いくつかのキーワードへの寄与度合のベクトルをスキーマ要素の意味とし，その類似性を定義している．適切なキーワードをどのように選んだらいいか，という問題がある．本章では，名称，構造からの類似度により類似スキーマ要素を発見し，それらの意味的関係の判定は設計者にゆだねるというアプローチをとった．また基数制約などの意味制約を概念グラフに変換する方法も考えられる．これらの意味の類似性まで加味した類似度で類似スキーマ要素を発見する方法とその効果は今後の研究を待つところである．

3.8 まとめ

スキーマ統合作業の中でも大量のデータ処理を必要とし，特に設計者の負荷の軽減が求められていた類似スキーマ要素の発見を中心に統合作業を計算機により支援する手法を提案した．この方法は，次の 3 点が特徴である．

- 名称，構造の観点からスキーマ要素間の類似度を求めること．
- 名称類似度の計算を容易にするため，属性名などをデータ標準化手法により標準化すること．
- 構造類似度の計算を容易にするため，共通データモデルとしてモデル化構成要素の少ない概念グラフを用いること．

この方法を実現するための要素として，標準化によって整備された属性名などの構文構成要素を利用することで類似度計算の対象となる概念を適切に取り出す概念グラフへの変換法および得られた概念グラフでのスキーマ要素間の類似度を標準化の結果を利用して計算する方法を提案した．

以上を踏まえてスキーマ統合支援ツールを作成し，これを用いた統合手順を確立した．例題を用いたツールの利用実績から，スキーマ統合に伴う稼働削減効果を推定し，約 1/3 へ削減できる可能性を示した．

また，グラフデータベースに対しても，その ER スキーマを生成することにより概念グラフへ変換できることを示し，その生成ルールを明らかにした．それによりグラフデータベースも．既存のデータベースのスキーマと同様に統合することが可能となった．

今後は実際のグラフデータベース等に適用して，実現性と効果を検証し，手順とツールの改善をすすめることが課題である．

第4章　動的に異種性解消するデータ統合検索技術を利用した制約つきグラフ探索

4.1　導入

本章では，制約つきグラフ探索を異種データベース環境で実現するための基本技術となるデータ統合技術を提案する．

グラフ情報の利用の増加に伴い，グラフ情報と既存のデータベース情報を組み合わせて利用する要求は，近年大きくなっている．しかし，そのときに既存のデータベース情報をグラフデータベースに移し変えたり，既存のアプリケーションに手を加えることは最小限に抑える必要がある．しかし，その連携のためには，情報源の持つデータの構造，名称，表現形式等の違い・異種性を解消し，データ統合検索を実現する必要がある．

そのような異種性を解消し情報源を統合するシステムとして，我々は事前にデータベースとユーザに見える項目の変換ルールを静的に定義しておくのではなく，グラフ化された異種性に関する断片的なメタデータを探索して組み合わせ，動的にユーザが利用するスキーマを構築し，データ統合検索できる技術を提案する．

まず，関係データベース等への適用を主眼とした基本的な技術を説明し，データ統合検索を構築するコストを削減できることを示す．

次に，それを制約つきグラフ探索に拡張する．Web 情報源機能を利用した制約つきグラフ探索の仮想化行い仮想的な表として実現する．これにより，他の情報源との基本的な統合が可能となる．さらに，制約つきグラフ探索の処理を階層的に分解することにより，探索処理をプッシュダウンする最適化手法を提案する．

4.2　技術課題と従来技術

複数のデータベースを統合するような検索を行うためには，以下のような情報源の異種性を解消する必要がある [A56][A42].

構造的異種性　各情報源のスキーマの構造の違いである．例えば，ある情報を一つの表で現したり，複数の表を用いて表す場合である．また，ER モ

デルにおいて実体で表現されたり，属性で表現されたりというモデルにおける表現要素の違いや，関係（RDB）で表現されていたり，木構造で表現されていたりする（XML）というモデルそのものの違いもこの分類に含める場合がある．

命名異種性 情報源のスキーマ要素の名前の違いである．例えば,「価格」と「値段」のように同じ意味を持つが名前のつけ方が違う場合である．

表現異種性 データの値の表現の違いである．例えば,「1m」と「100cm」のように単位が異なったり，価格を表すのに，ドルと円が異なったりという場合である．

このような異種性を解消し，情報源を統合検索するシステムとして，連邦データベース [A57] や，メディエータ [A58][A59] 等が提案されている．しかし，これらのシステムでは，基本的に，統合されたスキーマを事前に静的にSQL や論理言語等を用いて構築する必要があり，そのスキーマ統合のためのコスト（稼働）が大きいことが課題となっていた．また，統合するための異種性の定義の再利用性という観点の研究はなかった．

4.3 動的に異種性解消するデータ統合検索技術

本節では，スキーマ統合により構築されたメタデータを探索し，断片的な異種性の定義を動的に組み合わせて，異種性を解消しデータ統合検索する技術を提案する．技術の概要を述べ，構造的異種性・命名異種性・表現異種性を解消する手法をそれぞれ述べる．最後に制約つきグラフ探索への適用で利用・拡張される，情報源の能力を考慮した問合せ最適化について述べる．

4.3.1 技術の概要

技術の概要を図 4-1 に示す [B82][B81]．利用者は検索したい概念と概念に対する条件のみを指定し，その概念がどの情報源からどのように得られるかを指定しない．データ統合技術は，問合せ要求から，その要求する概念が存在する情報源を特定し，概念間の繋がりを補完し，問合せの処理順番を決定する「問合せ変換機能」と，利用者が選択した問合せ候補を実行し検索結果を得る「問合せ実行機能」の 2 つの機能から構成される．様々な情報源は仮想的な表としてモデル化され，問合せ処理も関係モデルにおける操作によって処理される．つまり共通データモデルとして関係データモデルが採用されている．Web 情報・XML 等の情報源は仮想的に関係データモデルにマッピングする．Web 情報や XML が持っている階層関係が失われるというデメリットはあるが，本技術の特徴である断片的な異種性解消定義と親和性が良く，デー

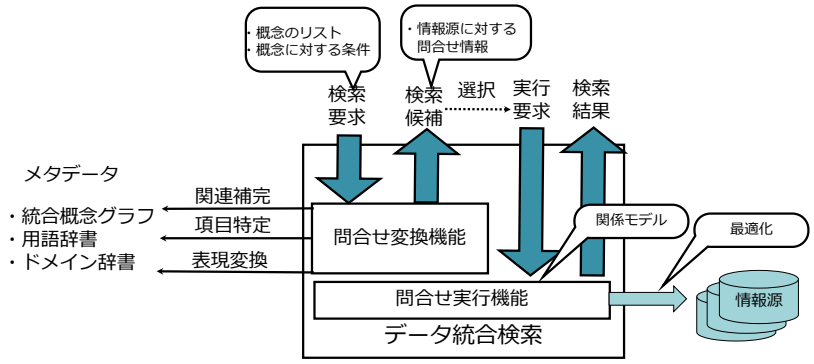


図 4-1: データ統合検索技術の概要

タ統合検索を単純で整理されたモデルによって実現でき，統合を容易に定義できるメリットがある．

問合せ変換機能は，指定された概念から，用語辞書を利用し対応する情報源のデータ項目を特定し，統合概念グラフを探索することにより概念間の関係を得て，ドメイン辞書から利用者側の表現形式と情報源の表現形式の関係から変換ルールを決定する．概念間の関係は，必要に応じてテーブル間やデータベース間を結合する問合せに変換される．

利用者は，適切な問合せ候補を選択し，データ統合検索を実行する．問合せ実行機能は，問合せ候補に含まれる情報を解釈し，複数の情報源から検索したデータを統合し，表現形式を変換して返却する．複数の候補を選択した場合，それぞれの検索結果の和集合が結果として返却される．

図 4-2 は，グラフデータベースを含む 3 つのデータベースから，統合検索を行う例であり，利用するメタデータとの関係も示している．統合概念グラフの探索，ドメインの変換，命名異種の解消等を行い問合せ候補を作成し実行している．

4.3.2 統合概念グラフ探索による問合せ候補生成

問合せ候補の生成には，前章のスキーマ統合手法によって作成された統合されたスキーマの概念グラフである統合概念グラフを利用する．まず，ユーザが指定した概念を発見し，統合概念グラフを探索し，指定された概念をすべて通るパスを見つける．スキーマ統合の結果として得られている概念とデータベースのデータ項目との対応関係を利用してアクセスすべき DB・テーブル・データ項目を発見する．そして，その繋がりからデータベースに対応する問合せと，データ統合処理実施側の問合せ処理の組み合わせを生成する．その例を図 4-3 に示す．この例では，社内の組織と社内サークルの関連を知る

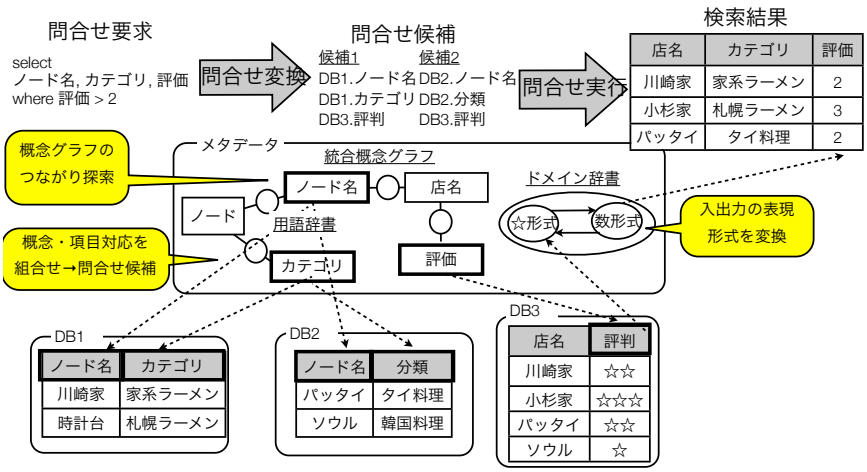


図 4-2: 動的なデータ統合検索の例とメタデータ

ために，組織_名称と社内サークル_名称をユーザが指定している．統合概念グラフを探索し，2 つのデータベースのテーブルを取得し，従業員_氏名と社内サークル員をキーとしてデータ統合処理実施側で 2 つのテーブルを結合し，所望の結果を得ることができる．

ここで，統合概念グラフ探索の方法は，最短経路探索ではなく，すべての経路を網羅的に候補としてあげるようにしている．実際，データベースの参照関係は，2 つのテーブル間に複数の関連を持つ場合もあるし，遠回りしてつながる場合もあり複数候補が出る場合が多い．このような候補は，異なる経路は最短ではなくとも，何らかの意味を持つ可能性があり，ユーザからの選択からはずすべきではないからである．本技術では，すべての組み合わせをユーザに返却し，ユーザの判断に委ねるという方針を取っている．

問合せ候補は，以下の方針で作成される．

- 可能なかぎり単一情報源に閉じる．
- 可能なかぎりジョインを行わない．
- ユーザから指定された項目を必ず含む（または多く含む）．

これらの方針によって，ユーザ要求からの適合率により問合せ候補はランキングされる（ランキングの詳細は [B81] 参照）．ユーザは問合せ候補から自分の所望の結果を選択し検索を実行できる．問合せ候補の複数選択はユニオンの演算として処理される．また，問合せ候補はビューとして保存することも可能である．

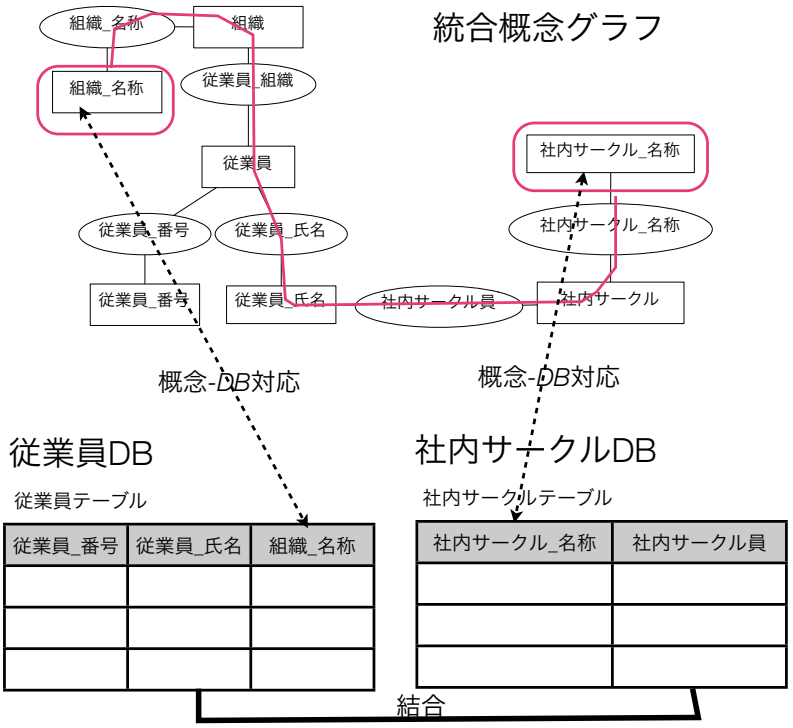


図 4-3: 統合概念グラフを利用した問合せ変換

4.3.3 用語辞書による命名異種性の解消

命名異種性の解消には，データ項目に関する用語辞書を利用している．ユーザ側に見える概念名と情報源側のデータ項目名を対応させている．用語辞書は，前章の概念グラフによるスキーマ統合において構築されている前提であるが，後に述べる本技術のシステム化で実現したメタデータ自動収集機能では，情報源のデータ項目を概念名とする初期状態を自動生成することも可能とした

用語辞書のイメージを図 4-4 に示す．例えば，情報源 1 のデータ項目「給料」と，情報源 2 のデータ項目「給与」がある場合，初期状態では,「給料」と「給与」がユーザに見える概念として登録される．用語辞書として，概念である「給与」と情報源 1 のデータ項目「給料」の対応を登録すると,「給与」という検索要求に対して，情報源 1「給料」と情報源 2「給与」の 2 つが問合せ候補としてマッチすることになる．

この用語辞書の採用はシンプルな技術ではあるが，統合スキーマの構築が単純なユニオンで済む場合は，用語辞書への登録のみで統合作業が終わることになる．これは SQL で情報源の対応をユニオン式等を用いて登録するより

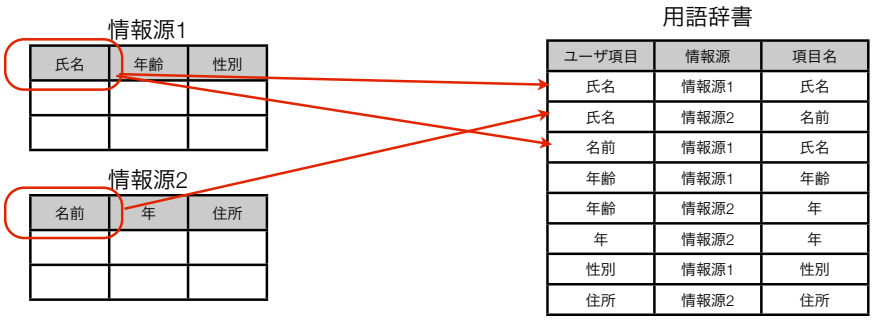


図 4-4: 用語辞書のイメージ

もかなり作業量を削減することができる．さらに，統合スキーマが固定的ではなく動的であるために，個々の用語の対応や異種性の解消の再利用性が増すと考えられる．本技術の問合せ候補の生成という機能が，一種のユーザによるビュー作成の支援となっており，断片的な用語対応との組み合わせでスキーマ統合の生産性を向上させることが期待できる．

4.3.4 動的な表現形式変換

表現異種性の解消には，ドメインという概念を用いる．ドメインは一般的なデータベースや情報資源管理で用いられる表現形式や型を表す概念に近いが，本技術にあわせた考え方の整理がされている．表現形式変換の最も単純なイメージを図 4-5 に示す．ユーザ向けの表現形式が事前に決まっていて，情報源とそれが異なっていれば，それを変換してユーザに返却し，ユーザからは一様な形式として見ることができる．

まず，情報源側の表現形式をメタデータとして管理する．これをローカルドメインと呼ぶ．次にユーザが利用する表現形式をメタデータとして管理する．これをユーザドメインと呼ぶ（このことから本技術にはユーザの概念が異種性解消のために必要となっている）．ドメイン間の変換関係は，表現形式をノード，表現形式間の変換関数をエッジとするグラフとしてモデル化できる．これらのドメインとその変換関係の情報は，前章のスキーマ統合によってドメイン辞書というメタデータとして構築されているので，それを本技術で取り込む．

表現形式の変換は以下のように処理される．まず，検索要求からユーザがほしい項目に対応する情報源の項目が特定され，そのローカルドメインがわかる．そのローカルドメインとユーザドメイン間の変換が定義されているかを表現形式グラフを用いて探索する．探索して見つかった場合，問合せ候補として返却する（これは表現形式の変換が可能でない場合は検索できないと

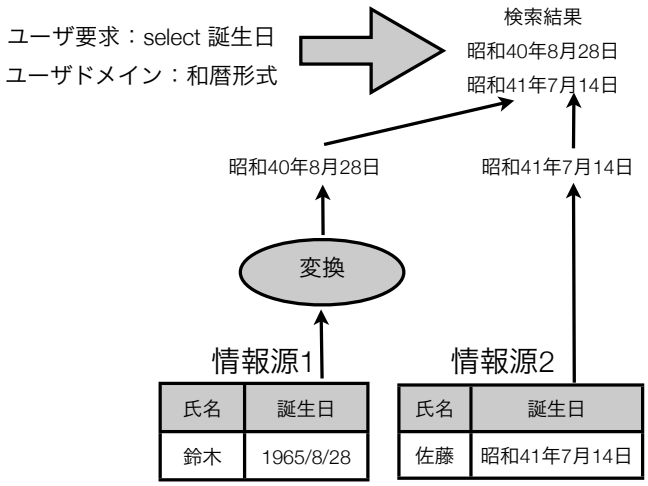


図 4-5: 表現形式変換の一例

いう考え方であるが，実際に開発したシステムではこの条件を緩和し，可能でない場合はその旨を通知するが候補とすることも可能となっている.）．検索実行の際は，探索されたエッジ上の表現形式変換関数を適用して，データを変換しユーザに返却する．探索が，複数のエッジを通過する場合は，複数の変換関数を組み合わせて実行される．

実用上は．多くのエッジをたどるような変換はあまり現れない．そのため，本技術を実用化したシステムでは，グローバルドメインという代表的なドメインを定めておき，そのドメインとの変換関数を定義することを推奨している．こうすると，表現形式変換グラフはグローバルドメインを中心とするスター上のグラフとなる．表現形式の変換はグローバルドメインを介した 2 段階変換でほとんどが実行される（グローバルドメインは，例えば，金額単位等のように表現形式の集まり単位で定義する，この集まりをドメイングループと呼んで管理している.）．

表現形式グラフの例を図 4-6 に示す．円がドメイングループ，四角がドメインで，ドメイン間の矢印が変換関数，太線四角がグローバルドメインを表す．変換関数の向きは表現形式の変換の向きを表している．すると，例えば，ユーザドメインがコード 3 で，ローカルドメインがコード 2 である場合，変換関数を探索し，コード 2 →コード 1 →コード 3 という結果が得られるので，この順に変換して，ユーザに返却する．この例ではユーザドメインがコード 2 で，ローカルドメインがコード 3 であれば探索が失敗する（コード 3 →コード 1 のエッジがない）ため，変換は不可となる．このような場合は，検索の最終候補からはずされる．

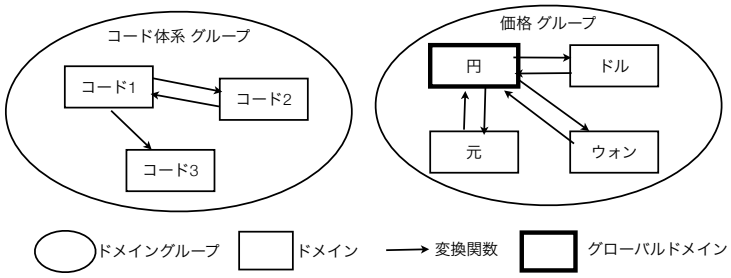


図 4-6: 表現形式グラフの例

4.3.5 情報源の能力を考慮した問合せ最適化

Web 情報・XML データベース・画像データベース等の様々な情報源を扱う場合，データベース側の検索能力がまちまちである場合がある．例えば，Web 情報の場合，フォームに条件値を入れられる項目もあるが，条件としては指定できず取得することしかできない項目もある．画像データベースの場合，データベースによって指定できるパラメータが異なっていることもある．

そのような状況に対応するために，情報源の能力を考慮した問合せ最適化を実施する．指定できる能力は，select 指定可能（値返却のできる項目），where 指定可能（条件として指定のできる項目），ソート指定可能（ソートキーとして使える項目）等がある．

また，画像検索等のユーザ定義関数を含む問合せを適切に生成するために，必須項目による制御を行う．ある情報源の仮想テーブルを検索する場合に，必ず条件として指定する必要のある項目を意味する．指定がない場合はデフォルト値を条件として付加することも可能である．これらの情報源能力の制約を満たすもののみが問合せ候補を生成できる．

筆者らは，これらの手法を XML データベース・画像データベース等の統合検索に活用し有効性を示している [B83]．

4.4 動的に異種性解消するデータ統合技術の実現と評価

4.4.1 システムとしての機能と特長

これまで述べた断片的な異種性の定義を利用した異種性解消技術を実現した，MediPresto/M と呼ぶシステムを開発した．MediPresto/M は，市販の様々な RDB だけではなく，Web サイト，XML，画像データベース等のデータ統合検索も実現した．全体図を図 4-7 に示す．

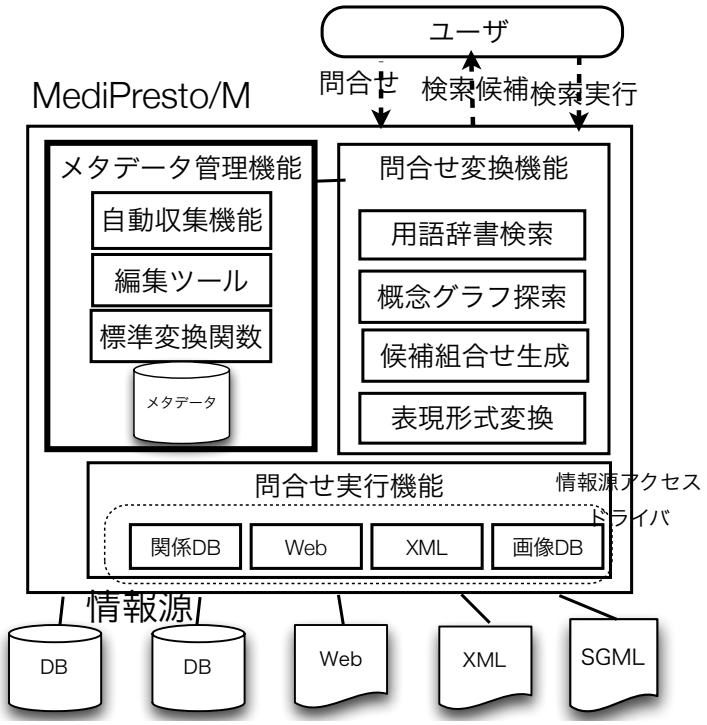


図 4-7: MediPresto/M の概要

MediPresto/M の実用上の大きな特長として以下の 3 点がある．

メタデータの自動収集機能 情報源のテーブル名・データ項目名等を自動的に収集し，MediPresto/M 上のメタデータとして構築できる機能を実現した．市販 DBMS のメタデータは，データ管理に比べて標準化できていない部分もあり，その収集は必ずしも簡単ではなかった．本自動収集を行うだけで，すぐに基本的な統合検索が可能な状態まで到達することができる．その状態から用語辞書定義，表現形式変換定義を組み合わせることにより実用的なスキーマ統合定義を作成することができる．

標準ドメイン変換関数 表現形式の変換関数として，頻繁に使われるものを標準関数として提供した，標準関数としては，データ流通プラットフォーム DB-STREAM で利用した標準関数を利用した．この標準関数に対してパラメータ（変換定数や変換表へのパス等）を設定することにより，変換関数を一から作成する必要がない．

メタデータ管理ツール GUI 上でメタデータや異種性定義を設定できるツールを整備した．上記の自動収集やドメイン変換の設定は一元的にこの

ツールを用いて実現している．統合の定義は従来は，SQL 等の言語をエディタ上で作成する必要があったが，本システムでは GUI 上での設定を行いつつ，実際の検索を実行しながら（Web からの統合検索ツールも提供している）調整することができ，異種情報源の統合の生産性を向上している．

MediPresto/M は，以下のようなサービスに向けて実用化され，その情報源統合の実用性が評価されている．

統合マルチメディアアーカイブサービス 大学にある図書館等の様々な情報源を統合したサービスに利用された（10 情報源）．

病院情報サービス 病院で医者の仕事リストを管理するサービスのための情報源統合に利用された（予約データベース，検査データベース，診断データベース）．

研究開発検索サービス 社内の研究開発の成果等に関する情報源の統合に利用された（1 データベース，1Web サイト，4SGML データベース）．

4.4.2 データ統合検索の構築稼働コストの比較評価

データ統合検索の構築とは，データ統合検索が行えるように，データ項目の対応付けを行ったり，表現形式変換関数を作成・設定したり，様々なメタデータを整備し，データ統合検索が行えるまでの準備作業のことを指すこととする．

我々は，複数の既存社内データベースシステムを基に，MediPresto/M 利用時．未使用時のそれぞれの場合で，データ統合検索の構築を試作し，その稼働を評価した．

既存社内データベースシステムの諸元を以下に示す．

- データベース数：5
- 総テーブル数：25
- 総データ項目数：627

比較結果を図に示す．未使用時に比較しておよそ 1/3 に削減することができた．削減の要因には，対応設計の容易さのようなデータ統合検索の方式に依存する要因と，表現形式変換関数やメタデータ設定ツールの整備のような MediPresto/M のシステムに依存する要因がある．特に，変換関数の整備のようなシステムに起因する要因はコスト削減に大きく貢献したが，設定稼働等の方式に要因する稼働にも削減効果があった（図 4-8）．

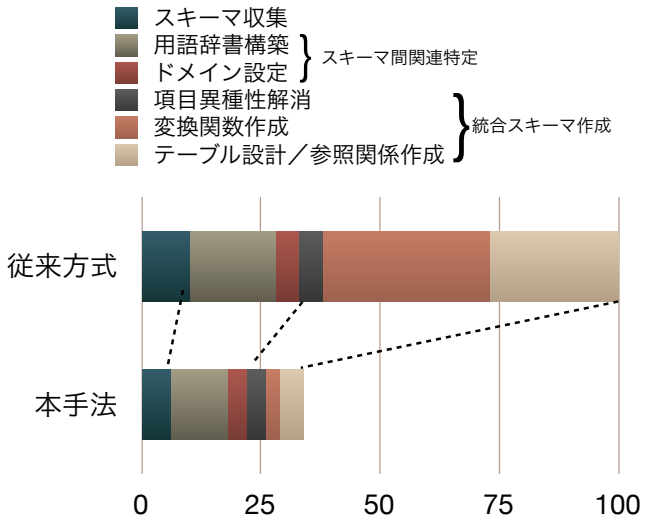


図 4-8: データ統合検索の構築稼働の比較

4.5 関連研究との比較

4.5.1 本技術の利点

この技術の利点は 3 つあげることができる．

動的な性質を利用したデータベース変更への追随の柔軟性 例えば，データベース上の 2 つのテーブルを性能向上のためにジョインして一つのテーブルとして管理するように変更した場合，本技術ではそのテーブルのメタデータのみ変更すれば，統合検索に対する影響がない．一般的な連邦データベース等のビュー定義であれば，そのビュー毎に変更に対する影響が発生していまう．

また，データベースのデータ項目の表現形式が変更された場合も，そのデータ項目のメタデータのみを変更すれば，(表現形式変換が事前に存在するという前提で) 統合検索に対する影響がない．これも，旧来の技術ではすべてのビュー定義に影響が発生する．

これらは，事前に静的にビュー定義をつくり込むのではなく，可能な限り動的な探索によって，異種性定義を組み合わせるという性質から実現できるものである．

異種性定義の断片化による再利用性向上 本技術では，このように表現形式の変換を動的に構成することができる．既存の技術では個別の表現形式の変換は統合ビューの定義に閉じてしまい，基本的には再利用ができない

表 4-1: 既存手法との比較

観点＼ビュー定義方式	SQL・XQuery	ルールベース	提案手法
動的・柔軟性	×	○ （実行時組合せ）	○
統合能力	○	○	△
定義の再利用性	×	○ （ルール単位）	○
必要スキルレベル	△	× （知識処理）	○

が（巧妙なビュー定義を行えば可能かもしれないが高度なスキルが必要である），表現形式の定義は断片的にされており，すべての統合ビュー作成で共用化される．関数の単位で再利用するのではなく，表現形式として再利用することがポイントである．

メタデータの進化 変換関数は動的に組み合わせられるために，定義を必要に応じて少しずつ作成していくことにより，ユーザは既存の変換定義を有効に活かしながら進化していくことができる．

4.5.2 本技術の位置づけ

本技術の特長は，前節で述べたように，多くの既存技術で採用されているSQL 等の言語による静的なビューを定義せずに，検索要求時に動的にメタデータを探索して，問合せ候補を生成することにある．この技術は，情報源の追加・スキーマ変更時の柔軟性が高い．メタデータは個々のビューに縛られておらず断片的であり，再利用性が高いからである．

本技術と既存の技術との比較を表 4-1 に示す．動的で定義の再利用性のある点で，本技術はルールベースでビューを定義する手法に類似していると言える．しかし，ルールベースのビュー定義では知識処理言語を利用する必要があり，統合記述能力は豊富なものの汎用的すぎて定義が困難である問題がある．本技術は，異種性解消に特化してメタデータの知識を構築する技術であり，ルールベースの技術に比べ，統合を記述する能力はやや劣るものの，定義が容易であることがメリットである．統合の記述能力に関する比較は次節で述べる．

本技術の利用者インターフェースは，概念のリストとその条件を指定するものであったが，これは，既存の研究としては，普遍関係をインターフェー

表 4-2: データ統合検索能力の比較

SQL の操作	実現	本技術での実現方法
select	○	候補生成時指定
where	○	候補生成時指定
distinct	○	候補生成時指定
order by	○	候補生成時指定
union	○	候補の選択
intersection	×	
minus	×	
join	○	関連を自動探索
outer join	○	関連エッジの種別で区別
group by	×	
subquery	×	

スとする手法と類似している．普遍関係とは，データベース中の全データが必ずその中に含まれる単一の仮想的な関係を言う．しかし，本研究と比較すると既存手法は不完全であると考えており．以下にその関係を論じる．

Zao ら [A50] は，製造業の複数社のデータベースを連携する方法を提案している．そのような環境ではデータベースの自律性が強く要求されることから，普遍関係を用いることにより緩やかな協調を図る方法を提案している．そこでは普遍関係の曖昧性の問題の解決のために，アクセスパス情報（結合演算等）や金額の単位等のスケール情報を文脈情報として利用する等の工夫がされていることが特長である．しかし，データ項目の変換機能はすべて利用者記述（ルールあるいは汎用プログラミング言語）であり，データ項目レベルでのスキーマ構築作業が筆者らの技術が筆者らの技術と比べて同等あるいは容易とは考え難い．

Reck ら [A51][A52] は，異種情報源アクセスへの透過的なアクセス（情報源の種類・所在などを奇にせずに済むアクセス）を可能とするためのアーキテクチャを提案している．その一部として，普遍関係に基づく利用者インタフェースを提案している．メディエータと呼ばれる意味的異種性を解消するソフトウェア（用語がやや混乱するが，情報源に対するラッパ相当であると考えられる）を複数構築する必要がある．この技術では普遍関係における曖昧性の問題の解決手段が示されていないことや，表現形式の変換において特別な工夫や標準的な関数の準備等はなく，容易にラッパを作成可能とは考え難い．

4.5.3　データ統合検索能力の比較

本技術はデータ統合検索環境の構築を簡易にすることに比べて，データ統合検索能力はやや犠牲にしている．表 5.2 は統合に使われる代表的な言語である SQL 言語における操作の，MediPresto/M における実装状況と，どのような実現方法をしているかをまとめたものである．select，where 等は候補生成時にユーザが指定する．union は候補を生成した後の選択として指定できる．join は自動的に探索される．outer join は join と同様に関連の探索だが join の関連と outer join の関連を異なるエッジとして区別し，違う候補として生成させユーザに選択させるという設計をしている．intersection，minus，group by，subquery 等は実装していない．これらを本技術に含めることは不可能ではないが，本技術の簡明性との親和性が悪く，定義の簡明性が失われてしまうこと，情報源の統合という観点で使われる頻度が少ないことという理由から，本技術には含めていない．実際には前に述べた適用例の実現で，これらの表現能力の制約が問題になったことはなかった．

4.6　制約つきグラフ探索への拡張

提案手法を制約つきグラフ探索に利用できるような拡張を行う．

4.6.1　Web 情報源による制約つきグラフ探索の実現

グラフ探索機能の仮想化と適切な問合せ候補生成の課題を解決するために，Web 情報源を利用したサービスの仮想化を利用する．よって，グラフデータベースへのアクセスは Web 経由であるという前提を置く．現在，最もポピュラーと考えられる Neo4j には REST API が用意されており，多くのグラフ探索サービスが Web 上から提供されているために，この前提は不自然ではない（セキュリティ上の課題はある）．

Web 情報源の利用例を図 4-9 に示す．問合せ要求に対し，検索条件を get，post のパラメータに変換し，Web ページの取得要求を行い，結果として得られる Web ページをテンプレートに従い表構造を抽出し，検索結果として返却する．

スキーマ統合の途中過程で得られたグラフデータベースの ER モデルの実体型（ノード，エッジ，機能）を，仮想的な表として，本手法の Web 情報源を定義する．そのとき，データ項目に対する制約を定義する．Web 情報源では，データ項目に入力のみ可能な項目，出力のみ可能な項目という制約がある．制約つきグラフ探索のパラメータである発ノードや着ノードは入力のみ指定可能であり，探索結果の経路は出力のみに可能な項目である．利用者の指定でこの制約を違反する場合は問合せ候補から除外する．ここまでは既存

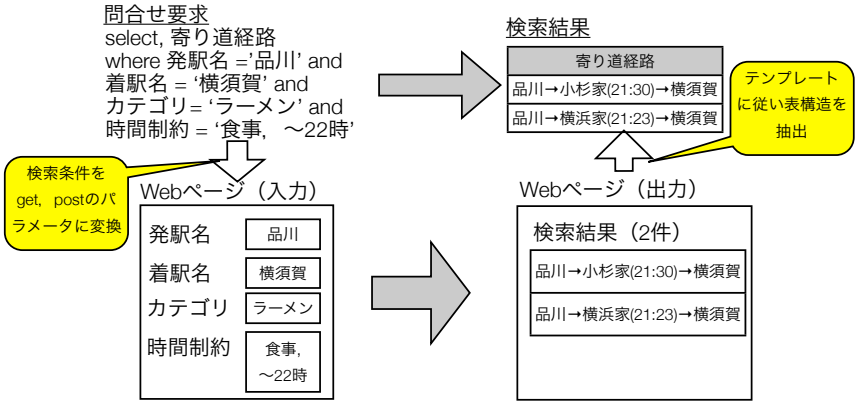


図 4-9: Web 情報源機能を利用したグラフ探索機能の仮想化

の Web 情報源技術で実現できる．この Web 情報源によって仮想化されたグラフデータベースと既存の情報源を組み合わせることにより，例えば，グラフデータベースに存在するカテゴリ情報より粗いカテゴリ情報を関係操作の結合により組合せ，他のデータベースにあるカテゴリ情報で寄り道探索を行うことが可能になる．また，複数のグラフ探索結果を関係操作の和として取得することも可能となる．

4.6.2 グラフ探索能力の階層化を利用した問合せ変換と最適化

前節の Web 情報源による手法で，制約つきグラフ探索機能の仮想化と問合せ候補生成は基本的には実現されているが，情報源側が必要なグラフ探索能力を完全に持つ必要があるという点で，情報源の能力に応じた問合せ最適化が実現されていない．情報源側にその能力がなければ問合せ候補から除外されてしまう．そこで，グラフ探索能力を階層的に定義し，可能な限り情報源側に実行させる技術を提案する．

まず，単純な例として最短経路を求めるダイクストラ探索を用い説明する．ダイクストラ探索は，エッジに移動コストの数値プロパティを持つグラフに対し，始点・終点を指定し合計移動コストが最小になる経路を求める探索である．グラフデータベース情報源として，ダイクストラ探索を持っていないケースを想定してみる．前述の Web 情報源による手法では，このグラフデータベース情報源にダイクストラ探索を実行することはできない．しかし，以下の条件が成立する場合を想定する．

- グラフデータベース情報源は，ノード・エッジ情報を取得する基本能力を外部に Web 経由で提供している．

- グラフデータベース情報源には，移動コストに相当するプロパティがエッジに存在している．
- データ統合検索機能側で，ダイクストラ探索を実行することができる．

この想定であれば，データ統合検索機能側でダイクストラ探索を実行し，グラフデータベース情報源にアクセスするときにノード・エッジ情報を取得する基本能力を利用すれば，ダイクストラ探索を機能として持たない情報源に対して，ダイクストラ探索を実行することが原理的に可能である．

この考え方を一般化し，ある制約つきグラフ探索があるとき，以下の仮定を置けば，情報源の能力の有無に関わらず，最終的に制約つきグラフ探索を実現できる．

- グラフデータベース情報源は，ノード・エッジ情報を取得する基本能力を外部に Web 経由で提供している．
- グラフデータベース情報源には，制約つきグラフ探索に必要なプロパティが存在している．
- データ統合検索機能側で，制約つきグラフ探索を実行することができる．

これは制約つきグラフ探索の能力の階層化の最も単純な例であり，最低レベルの基本能力と最高レベルの制約つきグラフ探索能力の 2 通りしかない場合である．これを階層的に処理を分解できる制約つきグラフ探索に一般化する．制約つきグラフ探索の処理分解を以下に定義する．

定義 3 (制約つきグラフ探索の処理分解) ノードプロパティ集合 P^V，エッジプロパティ集合 P^E に対する結果制約条件として C が与えられている制約つきグラフ探索に対して，以下の性質を持つ 2 つの制約つきグラフ探索がある場合，元の制約つきグラフ探索の処理分解と呼ぶ．ただし，$\circ$ の意味は制約の適用をパス集合に対して，続けて行う場合に，結果となるパス集合が等しいという意味とする．

$$P^V = {P_1}^V \cup {P_2}^V$$

$$P^E = {P_1}^E \cup {P_2}^E$$

$$C(P^V, P^E) = C_1({P_1}^V, {P_1}^E) \circ C_2({P_2}^V, {P_2}^E)$$

この分解の典型的な例は，制約つきグラフ探索の探索と制約のチェックが分離でき，探索後に制約チェックを行い探索結果を求められる場合である．しかし，ここで，結果となるパス集合が等しいという分解に関する成立条件は，厳しいものである．実際の問題で処理分解を考える場合，制約適用の可換性が一般には成り立たないため，分解した制約を重ねて適用した結果と，元の制約が一致しない場合が多い．しかし，そのような場合でも，実用上の精度要求から許容できることもあるので，パス集合の等しさは，その分解を定義

level	能力	必須プロパティ
3	x-ホップ内の趣味がyのサブグラフ内の取得	ノードid, エッジid, 趣味
2	x-ホップ内のサブグラフ取得	ノードid, エッジid
1	基本能力	ノードid, エッジid

趣味によるフィルタが差分能力

図 4-10: 例 1（SNS 探索）の能力階層と必須プロパティ

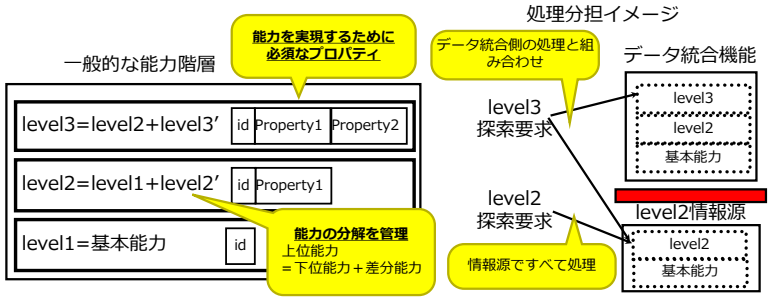


図 4-11: グラフ探索能力の階層化の管理

する（プログラミングする）開発者の判断に委ねられるとする．制約分解の数理的モデルによる一般化は，今後の課題である．

定義 3 は，ある制約つきグラフ探索がある場合に，その制約を，2 つに分解し，それぞれの分解で指定されるプロパティ集合は，元のプロパティ集合の部分集合となっていることを意味している．分解は階層的に定義されることが可能である．そのとき，分解によって，階層が下がるに従ってプロパティ集合は徐々に小さくなることになる．そして，最下層はプロパティを用いずに（プロパティが空集合の場合），id を指定してグラフから単純にノードとエッジを抜き出す最も基本的な処理が位置づけられる．

処理分解を用いた能力の階層化イメージを図 4-11 に示す．階層の高い能力は下位の能力を組み合わせて実現することができる．能力が高いほど多くのプロパティを指定した探索を行う．

図 4-11 は，グラフ探索能力の階層化の管理と，それを処理分担に反映させるイメージを示している．左図の最上位階層が意味することは，以下である．

- level3 能力は level2 能力と level3’ 能力に分解される
- level3 能力で利用するプロパティは Property1 と Property2 である

図 4-11 の下側の例は，k-top の最短経路探索をその近隣施設の値の条件でしぼり込む制約つきグラフ探索の例を示している．以下に，この考え方を用いて，データ統合検索におけるプッシュダウンを実現する方法を述べる．まず，次のメタデータを管理する．

- 制約つきグラフ探索処理と，その探索処理で利用されるプロパティの関係
- 制約つきグラフ探索処理の分解関係
- 情報源が実施できる制約つきグラフ探索処理

そして，このメタデータで管理される制約つきグラフ探索処理（分解されたものも含む）すべてはデータ統合検索の問合せ実行機能として実現されているという前提とする．上記メタデータを利用したデータ統合検索の問合せ変換機能の処理手順を示す．

- 情報源がどのレベルの階層まで処理できるかをチェックする．
- グラフ探索実行に必要なプロパティの存在を調べる．
- プロパティがすべて存在する場合は，情報源側で処理できると判断し，当該能力をプッシュダウンする問合せを生成する．
- プロパティが不足している場合は，次の下の階層の処理を情報源側で実施できるかを調べる．
- 以上のチェックを下位階層に向かって繰り返し実施し，最終的に情報源側の能力とプロパティが得られなかった場合は，最下層のレベル 1 の能力を用いてデータ統合検索側で制約つきグラフ探索を実施する．

以上の方式により，階層的な能力のプッシュダウン最適化が可能となる．

4.7 まとめ

メタデータのグラフ化を利用することにより，データの異種性を断片的に定義でき，動的に異種解消することのできるデータ統合技術を提案し，そのシステム化を行いスキーマ構築の稼働を評価し，約 1/3 に削減できることを示し，その有効性を示した．

さらに，データ統合検索技術を制約つきグラフ探索に適用できるように拡張した．制約つきグラフ探索の分解を定義し，その分解に基づく階層的な能力管理を利用して，プッシュダウンが可能な問合せ最適化技術を確立した．

今後の課題としては，以下の 2 つがある．

課題 **1**：動的特性の有効性の評価 動的な異種性解消を，データ統合の構築稼働という観点で評価したが，動的な特性は，スキーマ変更や情報源追加等に最も有効に働くと考えられる．その観点での評価は難しいが，今後，実施例の検証，シミュレーション等によって評価し有効性を示したい．

課題 **2**：制約つきグラフ探索のモデル詳細化 制約つきグラフ探索の分解を定義したが，分解した場合に同等な（厳密的な意味の同等と実用的な意味の同等がある）解が得られるかどうかは，分解を行う処理個別に開発者が判断し，実装する必要があった．制約つきグラフ探索を数理的にモデル化し，その探索結果の同一性を判定できれば，分解の存在の有無・分解の実装のために有効である．

第5章　異種データベース環境における時間制約つき寄り道探索の実現

5.1　導入

本章では，制約つきグラフ探索の一例として時間制約つき寄り道探索を提案し，単一データベース・分散データベースにおける適用を行い，その評価を行う．まず，時間制約つき寄り道探索を提案する背景と問題意識を説明する．

近年，カーナビゲーション，スマートフォン，地図サービス等の普及により，旅行計画作成に関するニーズが増大しており，様々なサービスが実用化されている．最も単純な旅行計画作成の例は，最短経路探索であり，鉄道網や道路網から所望のルートを探索するサービスに活用されている．これは，ノード間の移動コスト（例：所要時間）が与えられたグラフにおいて，合計コストが最小となる最短のルートを求める問題であり，ダイクストラ法，A* 法等のグラフ探索技術が確立している [A60][A61][A62].

一方，ユーザの位置情報に応じて，お勧めの場所を推薦するレコメンドサービスがある．ユーザの位置（GPS 等により取得）の近くにあり，ユーザの希望，趣向にあった店を推薦したり，目的地までのルートを探索したとき，そのルート沿いのガソリンスタンドやファーストフード店を推薦したり等，交通やグルメ等のインターネット上の検索サービスの分野 [A63][A64] や，カーナビ等のナビゲーションシステムで用いられている．

また，上記サービスに関係するが，少し目的が違う，次のような要求もある．例えば，ユーザがどこかに出張するときに，必ずしも最短経路でなく，多少遠回りしてもよいから，そのルート上で自分の好みの昼食をとってから目的地に到達したい，という要求である．この要求を満たす探索を「寄り道探索」と呼ぶことにする．このとき経由地を POI（Point of interest）と呼ぶ．ユーザからは，直接具体的な POI ではなく,「寿司が食べたい」とか「銀行に行きたい」というような行きたい場所のカテゴリで指定されることを想定する．具体的な POI が事前に決まっているのであれば，出発点-POI・POI-到達点の 2 つのルートを最短経路探索法で求めればよいので，技術的には既存の技術で解決できる．しかし，寄り道探索では，あらかじめわからない POI を探索しつつ，できるだけ最短経路に近いルートを求めることが必要になる．

寄り道探索において，寄り道先に時間制約がある場合を考える．例えば，開店時間には 12 時〜22 時のように制限がある．その開店時間に到着できるようなルートだけを結果としてほしい．例えば，夜遅めに出発するときには，閉店に間に合うように出発点に近い POI に寄り道する必要がある．また，開店時間だけでなく,「12 時〜13 時，ランチ 2 割引」や「17 時以降，閉店前 3 割引」のような時間限定のタイムセールのときに来店したいという要求も同じ課題であると言える．このような，時間制約をもつ寄り道探索を「時間制約つき寄り道探索」と呼ぶ．上記では，開店時間などのサービスの時間制約を例として挙げたが,「その店で 2 時間ゆっくりしたい」「銀行でお金をおろすのでぎりぎりに間に合えばよい」というような，ユーザにとっての時間制約の要求もあり，これらも満たすような探索を行いたい．

まず，時間制約を満たす解を導出する方法が課題となる．通常の寄り道探索では総所要時間は，出発点から POI までの所要時間と，POI から到達点までの所要時間を単純に足せばよいが，時間制約つき寄り道探索の場合，待ち時間も許容することがサービス的に有効である．待ち時間も含めたスケジュールを決定し，その総所要時間で最終的にランキングする必要がある．また，寄り道探索は POI を探すために，通常の最短経路探索以上に多くのノードを探索する必要があるため，できるだけノードの展開を減らし，探索速度を向上することも課題である．

大沢他 [A65] で提案されている寄り道探索のための逐次拡大法をそのまま利用し，それに時間制約つき寄り道探索解を導出する方法を組み合わせた基本導出法と，さらに時間制約を積極的にグラフ探索に利用する動的導出法を提案した．動的導出法は，単一データベースの場合，探索範囲と解候補の個数を制限し性能を改善できる．

さらに，時間制約つき寄り道探索は，交通網に関する情報とサービスに関する情報が，分散・異種になる可能性が高い特長を持つ．そこで，異種分散データベース環境における時間制約つき寄り道探索に，本研究の提案手法を適用し評価する．階層的な能力管理を用いた制約つきグラフ探索のプッシュダウン方式が，時間制約つき寄り道探索において，どのように実現されるかを示し，分散データベース環境の実験により評価し，処理を分解できる基本導出法が性能上有利になるケースが多いことを示した．

本章は以下のように構成されている．まず，5.2 節で過去の研究との関連を示し，5.3 節で時間制約つき寄り道探索を定義し，5.4 節で既存の寄り道探索の手法と基本導出法について述べる．5.5 節で動的導出法を示す．5.6 節で提案手法を単一のグラフデータベースを用いて実装した実験システムとその評価を述べ，5.7 節で異種分散データベース技術を用いた提案手法の適用について述べ，5.8 節で結論と今後の課題を述べる．

表 5-1: ルート探索手法の比較

	Sequenced Route 探索	時間制約つき寄り道探索
利用する距離	ユークリッド距離	グラフ上の距離
出発点指定	あり	あり
到着点指定	なし	あり
寄り道先の数	複数	1
順序性	あり	なし
時間指定	なし	あり
解の最適性	準最適解	最適解

5.2 関連研究

本章で扱っているのは，グラフ上で POI を探索し，条件付きの最短経路を求める問題であり，グラフデータベース・空間データベースの分野との関連が深い．空間データベース検索の分野では，k-最近傍探索等の分野で類似の POI を探索する方式が提案されている [A66][A67]．その代表例として，Sharifzadeh 他 [A66] による Optimal Sequenced Route 探索があげられる．これは，複数の異なる POI を指定された順序で探索するという一種の最近傍探索でありユークリッド距離を用いている．本章のようなグラフ上の最短経路探索ではない．また，Optimal Sequenced Route 探索は準最適解であることに対し，本章で提案する手法では最適解を求めることが可能である．表 5.2 にその違いをまとめた．一方，ユークリッド距離ではなくグラフ上の距離で，最短経路探索等を効率よく求める技術の研究が行われており [A68]，特にネットワークボロノイ図等を前処理で作成し効率化をはかる研究が盛んである [A69]．しかし，本章では頻繁に変更される時間制約があるような応用を想定する．例えば，我々が想定している応用として，レストランやスーパーマーケットのタイムセールの探索がある．これは，事前にスケジュールが決まっておらず，当日の在庫状況，顧客状況等によって実施が決定され，それを POI にいる従業員が twitter でつぶやくと探索の対象として登録されるというように，実時間性がある．その場合，前処理が必要であるとそのやり直しが頻繁に発生してしまうので適しておらず，前処理を必要としないことが望ましい．例えば，Sharifzadeh 他 [A66] や蒲原他 [A69] は，ネットワークボロノイ図を事前に作成しておく必要があり，時間制約を利用する問題に適さない．前処理を必要としない手法として，大沢他 [A65] による逐次拡大法がある．これは，POI が一つである制約はあるが，前処理なしで寄り道経路探索を実現している．本章ではその手法を拡張している．

時間の制約という意味では，時間概念を組み込んだ時制データベース [A70][A71] が研究されているが，関係モデルや XML 等の一般的なデータモデルに時間

概念を組み合わせる研究が主流になっている．本章は，グラフデータベースに時間概念を取り込んでいることに特徴があるが，サービスの概念に特化しているという意味で，一般的な時制グラフデータベースとまでは言えない．しかし，今後，一般的な時制グラフデータベースの研究を進める上での最初の一歩であり，重要な応用例になるものと考える．

本章では，ユーザやサービスによって指定された時間制約を満たす時間区間を求めているが，これは区間スケジューリング問題 [A72] に関連がある．区間スケジューリング問題では，要求する区間に重なりがなく，できるだけ多く（長く）要求を実行できる区間を選ぶことが目標になるが，本章では POI を一つに制約し，答えとなる時間区間の個数が三つに限られていることから，問題に特化した初等的な解法で時間区間を求めている．今後，POI を複数に拡張するときに，より一般的な区間スケジュール問題の成果を利用できる可能性がある．

ユーザの嗜好を反映したルートを提示するような研究としては，松田他 [A73] があるが,「歩道のある道を歩きたい」等の要求に応じて，コストに重みを設定しルートを計算する方法であり，本章で問題としているカテゴリに合致するような寄り道を行う探索には利用することができない．

グラフ情報を蓄積・検索する手段として，グラフのデータモデルを直接扱うことができるデータベース管理システムが一般的になりつつあり，グラフデータベースと呼ばれている [A74]．実用的に利用できる製品も近年でてきている [A75]．本章でも，システムの実現と評価に利用している．

5.3 時間制約つき寄り道探索

5.3.1 時間制約つき寄り道探索の定義

時間制約つき寄り道探索は，例えば，開店時間には 12 時〜22 時のように制限がある店に到着できるようなルートを求めるグラフ探索である．夜遅めに出発するときには，閉店に間に合うように出発点に近い POI に寄り道する必要があり,「その店で 2 時間ゆっくりしたい」「銀行でお金をおろすのでぎりぎりに間に合えばよい」というような，ユーザにとっての時間制約の要求もある．以下に定義を述べるが，これらの条件を満たす定義となっている．寄り道先は一箇所に限定している．$G = (V, E)$ を重みつきグラフとする．V が頂点（ノード）集合，E が辺（エッジ）集合である．エッジへの重みは移動にかかる時間を表すこととする．

定義 4 (カテゴリ) ノードの種別を表すラベルをカテゴリと呼ぶ．カテゴリが付与されたノードを *POI* と呼ぶ．カテゴリは，寄り道先として指定するために使う「コンビニ」「銀行」「中華料理屋」等の指定を意味する．

定義 **5** (時間区間) 時間区間を *[*開始時刻*,* 終了時刻*]* と表現する．以降の説明では，分単位で時刻と時間を表記し，時刻を表すときに年月日を省略する．

定義 **6** (サービス条件) *POI* においてサービスを識別するラベルをサービス内容と呼ぶ（例：営業時間，ランチサービス，商品 *2* 割引）．サービス内容とそのサービスが実施される時間区間の組をサービス条件と呼ぶ．一般的にはサービス実施時間はひとつの区間とは限らないが（例：昼と夜に営業），以降の解の導出法ではひとつのサービス実施は一つの区間であることを前提としている部分がある．一般化は今後の課題である，

定義 **7** (ユーザ条件) 以下の *5* つのユーザ側の時間制約をユーザ条件と呼ぶ．

- 出発点条件：出発点を出発し探索を開始する時刻を含む時間区間
- *POI* 開始条件：*POI* でサービスを開始する時刻を含む時間区間
- *POI* 終了条件：*POI* でサービスを終了する時刻を含む時間区間
- 滞在長条件：*POI* でサービスを受ける時間の長さの最大値・最小値
- 到着点条件：到着点に到着し探索を終了する時刻を含む時間区間

定義 **8** (時間制約つき寄り道探索) カテゴリ・サービス条件が与えられている重みつきグラフ $G = (V, E)$ に対して，時間制約つき寄り道探索とは，指定した出発点・到着点・カテゴリ・サービス内容・ユーザ条件・解の個数 k に対して，以下の条件を満たすグラフ上のルートと結果となる時間区間である S 区間（出発点から *POI* まで移動する時間区間），P 区間（*POI* に滞在する時間区間），E 区間（*POI* から到着点へ移動する時間区間）の組について，総所要時間（E 区間の最大値 - S 区間の最小値）が最小となる上位 k 組を求めることを言う．

- ルートは，指定した出発点を出発し，指定したカテゴリを付与された *POI* を経由し，指定した到着点に到着する
- S 区間の最小値は出発点条件に含まれる
- S 区間の長さは出発点から *POI* までの移動時間に一致する
- P 区間の最小値が，*POI* 開始条件に含まれる
- P 区間の最大値が，*POI* 終了条件に含まれる
- P 区間は *POI* における指定したサービス内容のサービス条件に含まれる
- P 区間の長さが，滞在長条件の最小値と最大値の間にある
- E 区間の最大値が到着点条件に含まれる

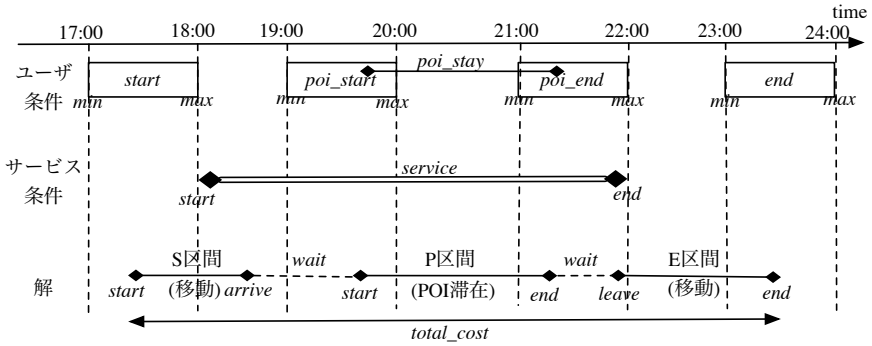


図 5-1: 時間制約つき寄り道探索の解

- E 区間の長さは POI から到着点までの移動時間に一致する

図 5-1 に，ユーザ条件・サービス条件とそれを満足する解の例を示す．時間制約つき寄り道は以下にあげる特徴をもち，単に時間制約を満たすルートを求めるだけでなく,「スケジュールを立案する」といった性格をもっている．

- 区間の位置は一意的に決まるとは限らない．P 区間は，この条件であれば，図 5-1 に示したように 19 時半〜21 時半でもよいし，19 時〜21 時としてもよい．移動時間が長ければ，それにより制約を受けるが，決定には任意性がある．その決定のためには，スケジュールに関する何らかの戦略（できるだけ早く物事を済ます，等）が必要となる．

- POI 滞在区間の前後の待ちを許容している．S 区間と E 区間が POI 滞在時間と重なってはいけない条件はあるが，S 区間の最大値は POI 開始条件に含まれる必要がないし，E 区間の最小値は POI 終了条件に含まれる必要がない．これは，例えば，POI がレストランとなるような応用において自然な仮定である．

ユーザ条件を細かく定義しているのは，できるだけ一般的な状況に対応するためであり，それらには実用上対応する意味がある．仕事終わりに友人と宴会をする場面で図 5-1 の値を例にして，以下に説明する（以降の説明でも POI への滞在を宴会の比喩で説明することがある）．

- 出発点条件の最小値：仕事が 17 時までなので，それ以前は出発できない

- 出発点条件の最大値：18 時に会社が閉まるので，その前に出発する

- POI 開始条件の最小値：友人が 19 時前にこれないので，それ以降に宴会を開始する

- POI 開始条件の最大値：宴会を遅くとも 20 時より前に始めたい

- POI 終了条件の最小値：21 時まではゆっくり宴会をしたい
- POI 終了条件の最大値：22 時以降は太るので宴会を終わらせる
- 到着条件の最小値：23 時にならないと家のカギが開かない
- 到着条件の最大値：門限が 24 時である
- 滞在長条件の最小値：2 時間は最低ゆっくり宴会をしたい
- 滞在長条件の最大値：長く滞在すると腰が痛くなり 2 時間が限界

また，チケット購入のように，窓口に滑り込みで間に合えばよいという場合，滞在長条件を 0 に設定すればよい．

今後の説明のために，ユーザ条件と解の区間の最大値と最小値に名前を付ける（図 5-1）．

- 出発点条件 $= [start\_min, start\_max]$
- POI 開始条件 $= [poi\_start\_min, poi\_start\_max]$
- POI 終了条件 $= [poi\_end\_min, poi\_end\_max]$
- 滞在長条件の最小値 $= poi\_stay\_min$
- 滞在長条件の最大値 $= poi\_stay\_max$
- S 区間 $= [total\_start, poi\_arrive]$
- P 区間 $= [poi\_stay\_start, poi\_stay\_end]$
- E 区間 $= [poi\_leave, total\_end]$

総所要時間を $total\_cost$，出発点から POI までの時間を $cost\_s$，POI から到着点までの時間を $cost\_e$ と表すこととする．以下の自明な関係が成り立つ．

- $total\_cost = total\_end - total\_start$
- $poi\_arrive = total\_start + cost\_s$
- $total\_end = poi\_leave + cost\_e$

時間制約つき寄り道探索のイメージを図 5-2 に示す．

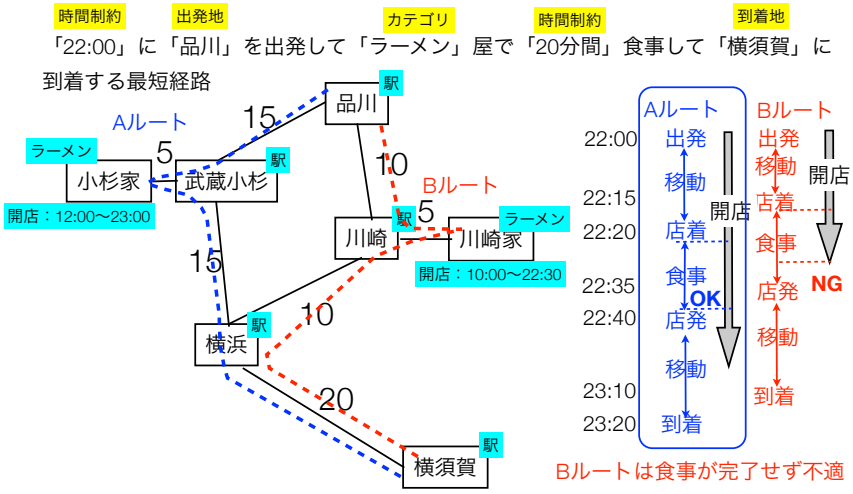


図 5-2: 時間制約つき寄り道探索

5.4 基本導出法

5.4.1 逐次拡大法による寄り道探索

基本導出法の基となる寄り道探索のための逐次拡大法（大沢他 [A65]）について説明する．基本的には，逐次拡大法は，双方向ダイクストラ探索をアルゴリズムとして利用し，出発点・到着点・POI の 3 点を結ぶ三角形に関する三角不等式を終了条件とすることが特長であり，最短経路探索の近年の研究 [A15][A16][A17] において利用された技術を応用的な探索に適用した手法である．その概要を図 5-3 に示す．

上位 k 個の解を求める逐次拡大法の基本的な手順を図 5-4 に示す．ダイクストラ法に相当する部分は簡略化している．出発点を S・到着点を E とする．基本的なアイディアは，出発点側（S 側）・到着点側（E 側）の両方からダイクストラ法を実行することである．指定したカテゴリのノードに到達したら，そのノード（これが見つかった候補 POI．複数ありえる）までのルートを候補テーブルに登録し，候補テーブルの中で，候補 POI に両側から到達するルートが登録されたら寄り道が完成したことになるので，それを繋げて結果テーブルに登録する．

繰り返しの終了条件であるが，通常のダイクストラ法ならば，到着点のコスト決定時点でよいが，寄り道探索では到着点ではなく POI を探すため，以下の条件とする必要がある．結果テーブルを RT，候補テーブルを CT，ルート R の全体コストを $cost(R)$，出発点（到着点）から POI までのコストを $cost\_s(R)$ ($cost\_e(R)$)，探索において次に展開するノードのコストを $cost\_s\_current$（S

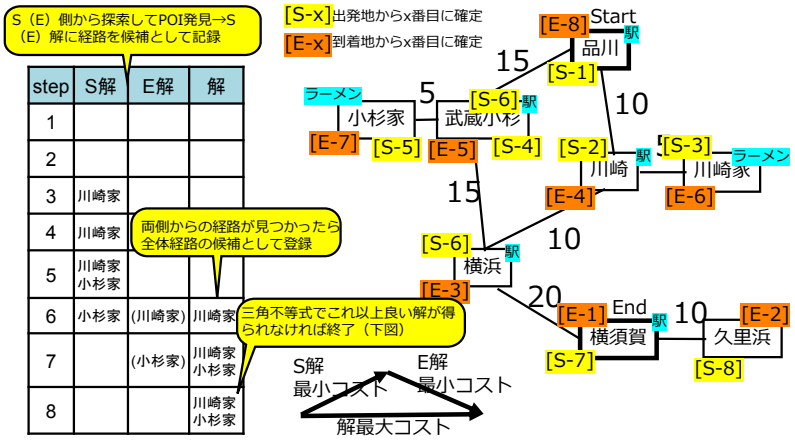


図 5-3: 逐次拡大法の概要

側），$cost\_e\_current$（E 側）とするとき，終了判定条件は，k 個の解が求められ，かつ以下の 2 つが両方成り立つことである．この式はよりコストの小さいルートが今後得られる可能性がないことを意味している．

$$max_{R \in RT}(cost(R)) \leq min_{R \in CT}(cost\_s(R)) + cost\_e\_current$$

$$max_{R \in RT}(cost(R)) \leq min_{R \in CT}(cost\_e(R)) + cost\_s\_current$$

ただし，POI 密度（全体ノード数に対する POI 数の割合）が低い場合，k 個の解が見つからないと，上記条件だけでは終了せずに延々と遠回りして解を探してしまう．よって，応用目的にあった現実的な値（例：レストラン探索なら 180 分以内）で上限を設けることも必要になる．

5.4.2 基本導出法の概要

基本導出法の概要を図 5-5 の左側に示す．まず既存の逐次拡大法を用いて，時間制約のない解を（最終的に求める個数よりも多めに）求める（非時間解と呼ぶ）．そして，非時間解が時間制約を満たすかを判定し，最終的な解（時間制約解と呼ぶ）を導出する．この方法のメリットは既存の寄り道探索手法をそのまま利用できることにある．一方，問題点は，非時間解を多めに求めたとして，時間制約解がその中に含まれる保証が必ずしもないことである．例えば，POI 密度が高いが時間制約を満たす POI が少ない場合，逐次拡大法は近くにある POI ばかりを非時間解として求めてしまって，与えられた時間制約を満たす POI に到達しない可能性がある．よって，非時間解は多めに求めておく必要があるが，その個数については 5.6 章で評価している．

while 展開すべきノード集合が空ではない **do**
  $N_S \leftarrow S$ 側の次の最小コストノード（ダイクストラ法）
  $N_E \leftarrow E$ 側の次の最小コストノード（ダイクストラ法）
  if 終了条件を満たす **then**
    return 結果テーブル
  end if
  if N_S がユーザ指定カテゴリを持つノード **then**
    候補テーブルに $S \rightarrow N_S$ のルートを登録
  end if
  if N_E がユーザ指定カテゴリを持つノード **then**
    候補テーブルに $N_E \rightarrow E$ のルートを登録
  end if
  if 候補テーブルに両側から同じ POI に到着するルートがある **then**
    見つかった S 側・E 側のルートをつなげて，結果テーブルに登録
    候補テーブルから，見つかった S 側・E 側のルートを削除
  end if
  結果テーブルを全体コストでソートし，上位 k 個のみを保持
  コストテーブルの更新（ダイクストラ法）
end while

図 5-4: 逐次拡大法のアルゴリズム（概要）

5.4.3 時間制約の判定

非時間解が時間制約を満たすかの判定について述べる．三つという限られた時間区間の判定なので，(a) 単独区間の制約判定，(b) 隣接二区間の制約判定，(c) 全体（三区間）の制約判定，を数直線上でチェックする初等的な方法でよい．また，方針としては，最初にサービス条件と POI 開始条件と POI 終了条件の制約をチェックする．それが満たされるのであれば，それらの交わりを新たな POI 開始条件と POI 終了条件としてその後のチェックに進む．このことで，実質的にサービス条件のチェックが終了するため，その後，サービス条件を考慮する必要がなくなる．つまり最初に P 区間単独の制約判定を行うことになる．

P 区間に関するチェックの例を図 5-6 に示す．最初に，POI 開始・終了条件とサービス条件に矛盾がないことをチェックする．POI への滞在開始時刻と滞在終了時刻はサービス時間に含まれる必要がある．以下に条件を示す．

- 条件 P-1：$service\_start \leq poi\_start\_max$
- 条件 P-2：$poi\_end\_min \leq service\_end$

次に，POI 開始・終了条件を補正する．これはサービス条件を POI 開始・

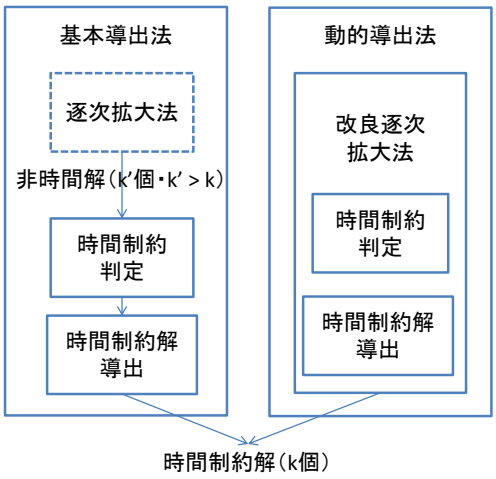


図 5-5: 基本導出法と動的導出法の概要

終了条件に反映させる処理である．POI 開始・終了条件とサービス条件で交わりを取り，それを補正された POI 開始条件・終了条件として今後の処理で使用する（図 5-6 の poi_start' と poi_end'）．

次に，補正された POI 開始・終了条件と POI 滞在時間に矛盾がないことをチェックする．図 5-6 に示すように，滞在時間が POI 開始・終了条件に対して短い場合と長い場合は矛盾である．制約を満たす条件を以下に示す．条件 P-3 が POI 滞在が短い場合の排除，条件 P-4 が POI 滞在が長い場合の排除である．これで P 区間単独としての制約チェックが完了する．

- 条件 P-3：$poi\_end'\_min < poi\_start'\_max + poi\_stay\_min$
- 条件 P-4：$poi\_start'\_min + poi\_stay\_max \leq poi\_end'\_max$

次に，S 区間と E 区間についての制約チェックを行う．非時間解で求まった出発点から POI までの時間（$cost\_s$）と POI から到着点までの時間（$cost\_e$）を利用する．図 5-7 にチェックの例を示す．まず，S 区間と E 区間の単独のチェックであるが，待ちを許容するから小さい分には制約はない（POI 開始条件（終了条件）内に到着（出発）する必要はない）ので，大きい場合のチェックのみでよい．S 区間は POI 開始条件を超えなければよく，E 区間は POI 終了条件を超えなければよい．以下に条件を示す．

- 条件 S：$start\_min + cost\_s < poi\_start'\_max$
- 条件 E：$poi\_end'\_min + cost\_e < end\_max$

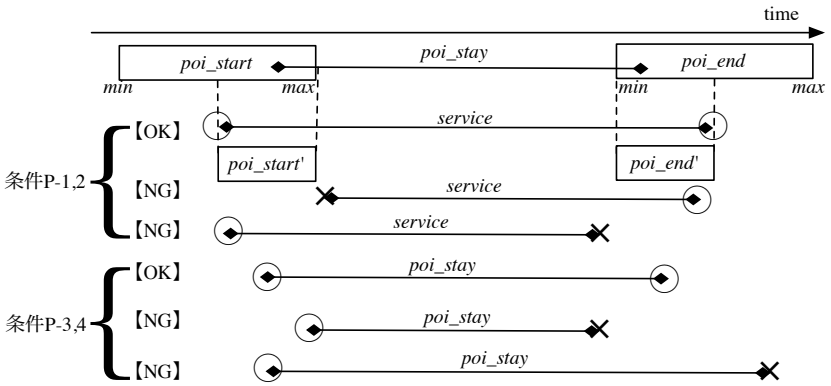


図 5-6: P 区間の制約チェック

次に，隣接二区間の制約判定であるが，それぞれ 2 つを繋げた区間が大きすぎて，POI 滞在終了・開始を超えるということがなければよい（小さい場合については，単独区間の制約のみでかまわない）．以下に条件を示す．

- 条件 S-E：$start\_min + cost\_s + poi\_stay\_min \leq poi\_end'\_max$
- 条件 P-E：$poi\_start'\_min \leq end\_max - (cost\_e + poi\_stay\_min)$

最後に三区間全体の整合であるが，出発から到着までのトータルの長さが区間の最大幅を超えなければ良い．以下に条件を示す．

- 条件 S-P-E：$cost\_s + poi\_stay\_min + cost\_e < end\_max - start\_min$

これで制約条件のチェックは完了である．

5.4.4 時間制約解の導出

次に，時間制約解の導出について説明する．前節のチェックによって，「時間（の長さ）」について矛盾するような条件は排除されているが，これまで与えられた制約だけでは，いつ出発し，POI に滞在し，目的地に到着するという「時刻」は一意には決定されない（総所要時間は一意に決まらない）．時刻を決定するためには，スケジュールの戦略を前提としておく必要がある．ここでは，代表的な「できるだけ待ちをなくし，かつ早く行動する」という戦略に基づく方法を提示する．戦略の特徴を以下にあげる．

- できるだけ早く POI 滞在を開始する

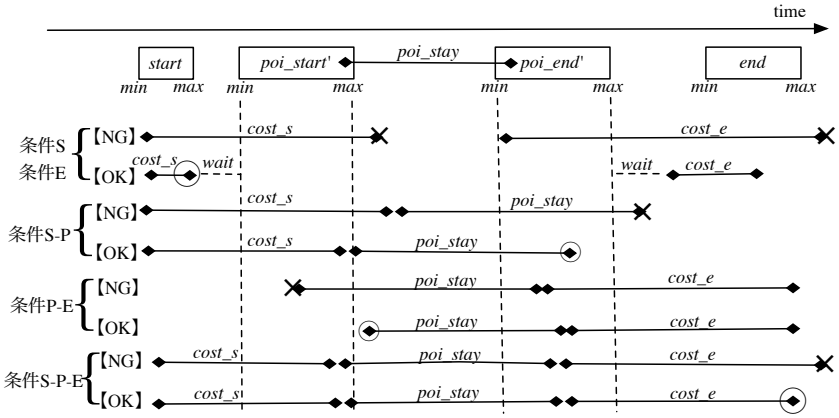


図 5-7: S 区間と E 区間の制約チェック

- できるだけ早く帰宅しようとする
- 待ち時間はできるだけ発生させない
- POI でのサービスに間に合う範囲で、出発点の出発時間を遅らせる

この戦略は単純化されており，現実の適用領域によっては適さない場合もあるが，それは下記に述べる方法を基本として修正すれば実現できる．

以下の導出では単純化のため P 区間の長さは固定であると仮定する（この値を poi_stay とする）．

時間制約解の導出は 2 つのフェーズによって構成される．

- 個別値決定フェーズ
- 調整フェーズ

個別値決定フェーズとは，全体としてのスケジュールの整合性は無視し，三つの区間の位置をそれぞれ戦略に合うように（この場合はできるだけ早めに行動するように）決定する．調整フェーズでは，前フェーズで決まった区間が重ならないように調整する．

図 5-8 に個別値決定フェーズのイメージを示し，決定するための場合分け条件を表 5-2 に示す．この表の条件ですべての値が決定される．この表に入らないような区間が長すぎるケース，短すぎるケースは事前に除外されている．

S 区間は，$cost\_s$ の値が小さいときは，許容されるぎりぎり（$start\_max$）に出発し，POI 到着後宴会開始まで待つことになる（Case1）．$cost\_s$ がだんだん大きくなり待ちが短くなり，poi_arrive が poi_start_min に到達すると，次に $total\_start$ が前倒しされる（Case2）．$total\_start$ が $start\_min$ に

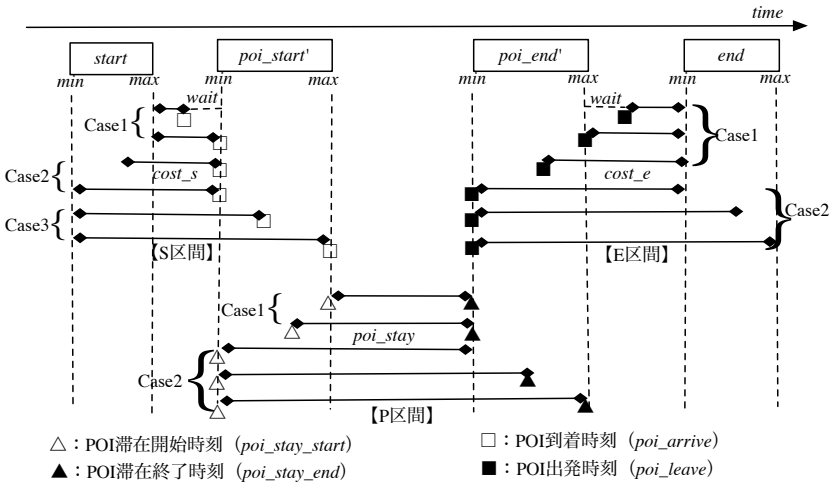


図 5-8: 個別値の決定イメージ

到達した時点で，poi_arrive が後ろにずれていく（Case3）．$cost\_s$ の最大値は $poi\_start\_max - start\_min$ である．

P 区間は，POI 滞在時間（poi_stay）が小さいときは，許容されるぎりぎり（poi_end_min）に終了するように宴会が行われる（Case1）．POI 滞在時間がだんだん大きくなり，poi_stay_start が poi_start_min に到達すると，次に poi_stay_end が後ろにずれる（Case2）．POI 滞在時間の最大値は $poi\_end\_max - poi\_start\_min$ である．

E 区間は，$cost\_e$ の値が小さいときは，許容されるぎりぎり（end_min）に到着するように POI を出発する．宴会終了後 POI 出発まで待つことになる（Case1）．$cost\_e$ がだんだん大きくなり待ちが短くなり，poi_leave が poi_end_min に到達すると，次に $total\_end$ が後ろにずれる（Case2）．$cost\_e$ の最大値は $end\_max - poi\_end\_min$ である．

次に調整フェーズについて述べる．上記パターンの組み合わせによって全体が決定されるが，調整が必要な場合として，一般的には以下の 2 つがある．

- 各区間の間に空きがある場合
- 各区間に重なりがある場合

しかし，POI 滞在時間を固定としたために，空きがある場合は，それは待ち時間となり，調整の余地はない．重なりがある場合の解決方法であるが，イメージを図 5-9 に示す．個別値決定では可能なかぎり，前倒しになるようにスケジュールが決定されているので，重なりが発生した場合は，後の区間を後ろにずらせばよい．S 区間と P 区間が重なったら，P 区間を後ろにずら

表 5-2: 個別値の決定

場合 (Case)		条件	変数	値	備考 (意味解釈)
S 区間	1	*cost_s =< poi_start_min - start_max*	*total_start*	*start_max*	往路が短く待ち発生
	2	*poi_start_min - start_max < cost_s* *cost_s <= poi_start_min - start_min*	*total_start*	*poi_start_min - cost_s*	待たずに最も早く宴会開始
	3	*poi_start_min - start_min < cost_s* *cost_s <= poi_start_max - start_min*	*total_start*	*start_min*	往路が長く宴会が後ろにずれる
P 区間	1	*poi_end_min - poi_start_max <= poi_stay* *poi_stay < poi_end_min - poi_start_min*	*poi_stay_start*	*poi_end_min - poi_stay*	宴会は最も早く終る
	2	*poi_end_min - poi_start_min <= poi_stay* *poi_stay <= poi_end_max - poi_start_min*	*poi_stay_start*	*poi_start_min*	宴会が長いため終了が後ろにずれる
E 区間	1	*cost_e =< end_min - poi_end_min*	*poi_leave*	*end_min - cost_s*	最も帰宅が早い
	2	*end_min - poi_end_min < cost_e* *cost_e <= end_max - poi_end_min*	*poi_leave*	*poi_end_min*	復路が長く帰宅が後ろにずれる

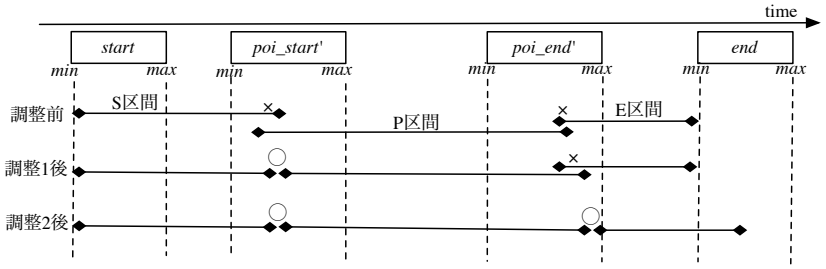


図 5-9: 調整フェーズのイメージ

し，さらに P 区間と E 区間が重なったら E 区間を後ろにずらす．その結果，$total\_end$ が end_max を超えるケースは事前に除外されているため正しい解が得られる．

5.5 動的導出法

時間制約つき寄り道探索は，時間制約を追加することにより，一般的にはグラフの探索空間が寄り道探索と比較して広範囲となる．その理由は，本探索の定義において，POI における滞在時間と待ち時間がモデル化されており，それを可変化できることにある．片側から探索を開始し POI に到達した場合，到着時点でサービスを受けることができずに待つ場合がある．その場合，その POI よりも遠くに，距離としては遠いが，サービスをすぐ開始できる POI を発見できる可能性がある．そのため，時間制約つき寄り道探索では，寄り道探索の終了条件を緩和する必要がある（基本導出法においては，それは非

時間解を多めに求めることにより対応されているが，次に述べる動的導出法では終了条件不等式に POI への最小滞在時間が加えられている.）

そこで，我々は，探索空間を狭めるために，時間制約を利用した枝刈り（temporal pruning）を行うことが特長である動的導出法を提案する．アルゴリズムの複雑さという観点では逐次拡大法とは変わらないが，交通検索等の実応用の条件によって枝刈りが有効となる．ユーザの時間制約とサービスの時間制約は，双方向ダイクストラ探索における両端からの探索ステップそれぞれでチェックされ，そのノードの先で解が得られないことが判明した時点でそのノードの展開を中止する．動的導出法は，時間制約条件を積極的に利用して，時間制約解を探索中に完成させ，事前に候補を絞り込み性能向上をはかることができる．図 5-5 の右側にイメージを示しているが，基本導出法で用いた時間制約判定と時間制約解導出を逐次拡大法の中に取り込んでいる．単純に取り込むだけではなく，できるだけ早めに絞り込みを行うための修正を行っている．

基本導出法における時間制約判定は，最初に P 区間のチェックを行い，POI 開始・終了条件の補正を行ったために，このままでは全ての条件が POI 確定後ではないとチェックできない．しかし，条件 S，条件 E，条件 S-P，条件 P-E は，補正前の POI 開始・終了条件を用いてチェックすることができ，POI 確定前にチェックできる条件である．以下に条件を示す（それぞれ元の条件を補正前に変更したものである）.

- 条件 S’：$start\_min + cost\_s \leq poi\_start\_max$
- 条件 E’：$poi\_end\_min + cost\_e \leq end\_max$
- 条件 S-P’：$start\_min + cost\_s + poi\_stay\_min \leq poi\_end\_max$
- 条件 P-E’：$poi\_start\_min \leq end\_max - (cost\_e + poi\_stay\_min)$

この 4 条件を含んだ 13 条件をできるだけ早い段階でチェックすることにより性能向上をはかる．その処理フローを，図 5-10 に示す（図の (b) は (a) における「S 側（E 側）で確定したノードが POI かチェック」の中のフローである.）．元となっている逐次拡大法に対する修正点を，図中で二重線で示している．以下の 3 つの契機で候補を絞り込む．それぞれでチェックできる条件を括弧内に書く．また，それらの契機の例を図 5-11 に示す.

- 契機 1：ノードへの最短距離が確定する時点（条件 S’, E’, S-P’, P-E’）
- 契機 2：片側から POI に到達する時点（条件 S, E, P-1〜4, S-P, P-E）
- 契機 3：POI への両側からのパスが見つかる時点（条件 S-P-E）

契機 1 のチェックは，探索範囲の限定に効果がある．契機 2 のチェックは，解の候補テーブルを小さくする効果がある．契機 3 のチェックは，最終的に全

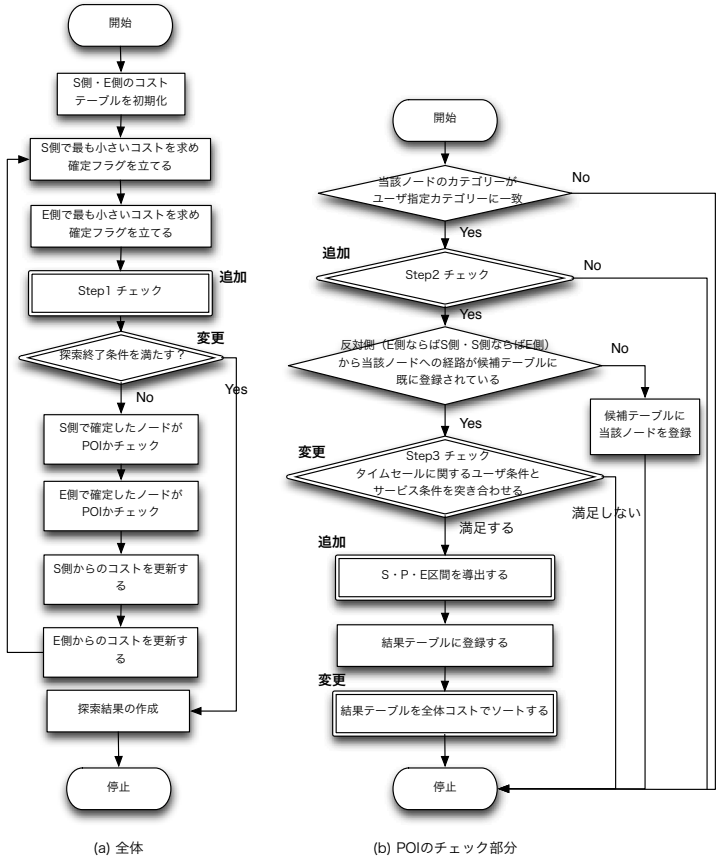


図 5-10: 動的導出法のフロー図

体としての制約を満たすため必要なチェックである．条件 S', E', S-P', P-E' は，POI に到達する前にチェックできるので，契機 1 でのチェックが可能である．チェックの結果 NG になった場合は，そのノードに至るコストを無限大に設定し，そのノードの先を展開しないようにする．一方，条件 S, E, P-1〜4, S-P, P-E は，POI に到達しないとチェックができないが，両側からのコストを要求していないので，契機 2 でのチェックが可能である．チェックの結果 NG になった場合は，候補テーブルへの追加を行わない．条件 S-P-E は両側からのコストが必要なので，契機 3 でのチェックが必要である．チェックの結果 NG になった場合は結果テーブルへの追加を行わない．

チェックが終わってから，S，P，E 区間を導出するが，これは基本導出法と同じ方法である．そこで求まった総所要時間によって，結果テーブルのソートを行う．

また，終了条件において，POI 滞在時間を考慮する必要があるため，4.1 節

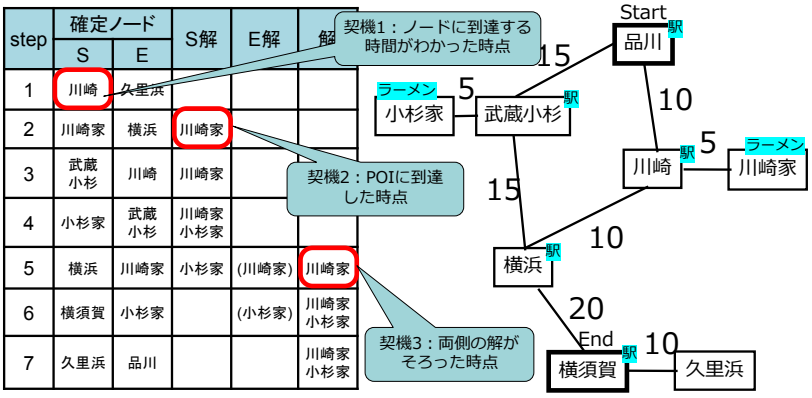


図 5-11: 動的導出法における探索の枝刈り契機

で述べた逐次拡大法における終了条件を以下のように修正する必要がある．

$$max_{R \in RT}(cost(R)) \leq min_{R \in CT}(cost\_s(R)) + cost\_e\_current + poi\_stay\_min$$

$$max_{R \in RT}(cost(R)) \leq min_{R \in CT}(cost\_e(R)) + cost\_s\_current + poi\_stay\_min$$

5.6 単一グラフデータベースにおける実験と評価

基本導出法，動的導出法を実装し評価した．グラフ情報を実装するために，グラフデータベースである Neo4j[A75] を利用した．首都圏の実際の鉄道網の情報と，POI 情報をグラフデータベースとして構築した（ノード数 64 万件，エッジ数 60 万件）．グラフデータの例を図 5-12 に示す．鉄道網に関しては，駅をノード，路線をエッジとして表現し，POI 情報はノードとして表現し，駅と POI の間は徒歩による移動時間を入れてエッジとして表現している．また，今回の実験システムでは，サービス情報と時間情報もグラフデータベース内のノードとして構築した．サービス情報と時間情報に対するエッジは，鉄道網と POI 情報のエッジと種別を区別することによって，グラフ探索の範囲から除外している．サービス情報と時間情報はできるだけ汎用的になるように設計している．一つの POI が複数のサービス（例：ランチサービス，宴会サービス）をもつ場合，それは別ノードとして表現する．時間情報はひとつの区間に対応しており，一つのサービスに対して，複数区間の営業時間がある場合（例：ランチ後に休憩あり），時間情報のノードを複数持たせることによって表現することができる．時間情報にはサービス実施，非実施の識別子があり，サービスを実施している区間とサービスを実施していない区間という表現もできる．

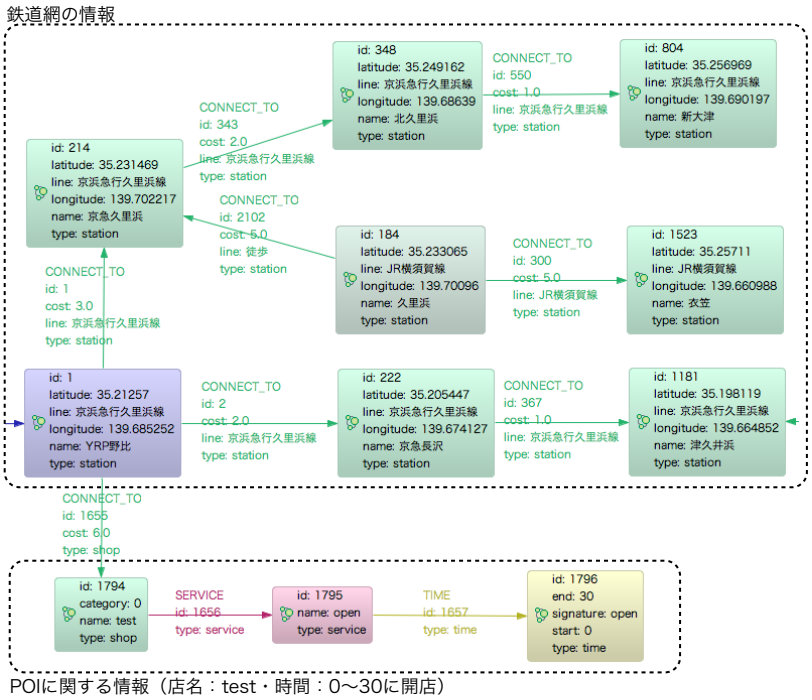


図 5-12: グラフデータベースのデータ例

このグラフデータベースに対して，時間制約つき寄り道探索の基本導出法，動的導出法を実装し，Web からもアクセス可能な，時間制約つき寄り道探索の実験システムを構築した（実験環境：Apple MacPro（Xeon 2.8GHz × 2），メモリ 12GB，SSD128GB，Mac OS X 10.6.7．プログラミング言語：Java（JDK 6））．その画面例を図 5-13 に示す．ユーザは出発地，到着地，寄り道先のカテゴリ，サービス内容，ユーザ条件となる時間制約を指定して探索実行すると，探索結果のスケジュールとルートを返却する．画面は実験目的であるので，すべての時間制約を入力可能にしているが，実際のアプリケーションでは，時間制約をスケジュールシステムから取得したり，個人毎に事前設定したりするやり方も考えられる．

まず，基本導出法において，中間解として生成される非時間解の個数をどの程度にすべきかを見積もった．データとしては上記の鉄道網データに加えて，サービス密度（交通網のノード数に対して，指定されるサービス内容を実施している POI の数の割合）を変化させ，評価用に自動生成した POI データを用いている（ノード数とエッジ数はともに約 4000 件）．また，POI は各駅に隣接して 1 つずつ設定した．よって，サービス密度が 100%とは探索するすべての駅毎にサービスが実行されているという意味となる．

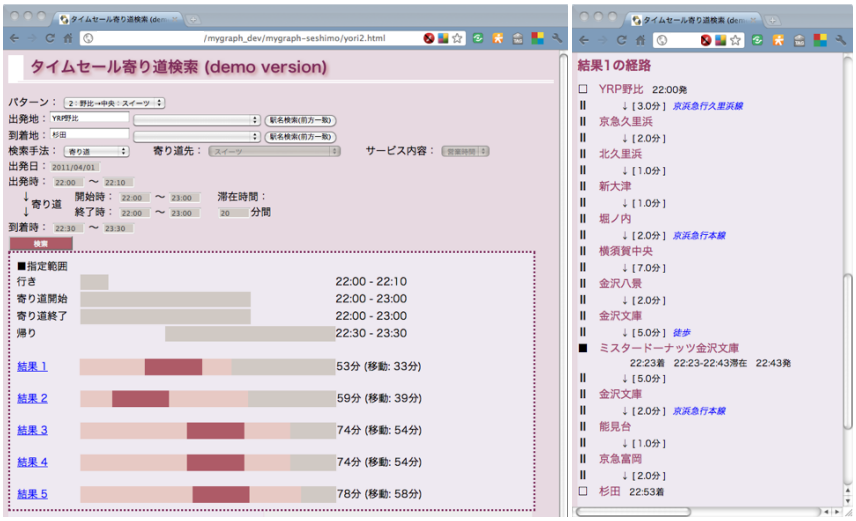


図 5-13: 寄り道探索システムの画面

我々は，本章の手法の適用範囲を，サービス密度が 20%以下の比較的サービス密度が低い場合を想定し，評価した．サービス密度が高い場合，本章で述べたような手法を用いずに，出発点と到着点の間の最短経路を求め，その経路上にある POI を列挙する簡易な手法でも，ある程度実用的な解を求められることが，その理由である（POI の分布が偏っていれば，正解を見つけられない場合はある）．しかし，サービス密度が 20%の場合とは，今回の評価データである鉄道網であれば，5 駅のうち 1 つでサービスを実施することに相当し，道路網においては，エッジの長さが信号間の距離 200m 程度であると想定すれば，約 1km 毎にサービスがあることに相当する（エッジが直線的に連続する場合）．これは，サービスへの適用範囲としては十分に広いと考えた．実際の実用的なサービスにおけるサービス密度の調査については今後の課題である．

まず，動的導出法によって正解となる 10 個の時間制約解を求めておいて，非時間解の個数を変化させて，同じ条件で基本導出法を実行したときに正解となる時間制約解を 10 個中何個導出できるかを調べた．評価結果を図 5-14 に示す．サービス密度が低くなる場合，非時間解を多く生成しないと十分な時間制約解を求めることができない．以降の評価では，最終的に返却する解の個数を 1，5 の 2 通りについて評価するために．5 個の正解を得ることができるように，非時間解の個数を 1%：500，5%：100，10%：20，20%：20，と設定した．

次に，基本導出法と動的導出法の探索を実行し，展開された総ノード数，候補テーブルの要素の最大個数，レスポンスタイムを測定した．ユーザの利用

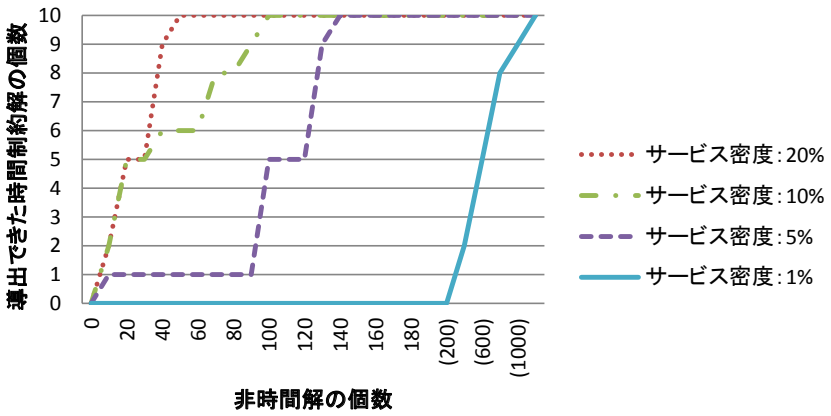


図 5-14: 基本導出法における非時間解の個数と時間制約解の個数の関係

シーンとして，以下を想定した．

- 移動途中の 1 時間の空き時間に，タイムセールで安くなったケーキをお土産として買う
- 買い物時間は 10 分
- タイムセールの時間は 1 時間

この利用シーンに相当する時間制約を与え探索を実行し，測定した結果を図 5-15，図 5-16，図 5-17 に示す（出発点，到着点を 2000 組ランダムに選び，その中で解が見つかったものの平均値を示している）．時間制約解の個数は 1，5 の 2 通り測定したが，基本導出法についてはどちらの結果もほぼ同じである．特にサービス密度が低い 1%，5%の場合に，動的導出法の結果がすぐれている．例えば，サービス密度が 1%の場合は，動的導出法を用いることにより，総展開ノード数が 34〜75%に，候補テーブルの要素数が 2〜7%に減り，レスポンスタイムが 11〜24 倍に改善している．特に時間制約解が 1 つの場合に改善効果が大きい．また，図 5-16，図 5-17 から，総展開ノード数と候補テーブル要素数が動的導出で大幅に削減されており，性能向上に貢献していると推測される．また，サービス密度が高い 20%の場合には，動的導出法の優位性は小さくなるものの，そのオーバーヘッドは大きくないことがわかる．今回の測定では，鉄道網というグラフとしては小規模な例で評価したが，道路網等のより大きいデータになれば，この差は広がるため，動的導出法は大規模グラフデータベースに対してはより有効であると考えられる．

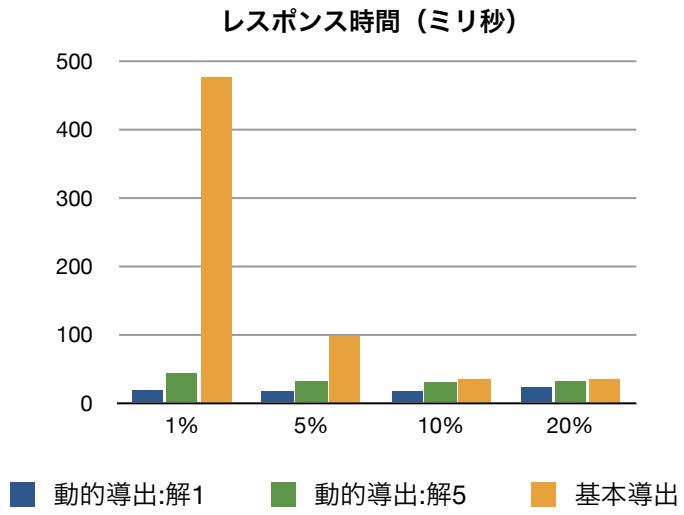


図 5-15: 基本導出法と動的導出法の比較（レスポンス時間）

5.7 異種分散データベースへの適用と評価

5.7.1 時間制約つき寄り道探索の能力の階層化

時間制約つき寄り道探索能力を階層化する．基本導出法は，時間制約のない寄り道探索と，制約のチェックに処理の組合せに分解できることから，能力は以下の 3 レベルに階層化できる．

- レベル 1 能力：ノード取得・エッジ取得の基本能力
- レベル 2 能力：時間制約のない寄り道探索
- レベル 3 能力：時間制約のある寄り道探索（基本導出法と動的導出法）

レベル 2 の必須プロパティは，エッジのコスト情報と，寄り道先の選択に利用するノードのカテゴリ情報である．レベル 3 の必須プロパティは，レベル 2 の必須プロパティに加えて，ノードの時間制約（開店時間等）情報となる．また，エッジのコスト情報が，距離ではなく時間長である制約もある．

寄り道探索は，両端からのダイクストラ探索を利用しているため，ダイクストラ探索を階層として設定できそうだが，単純な組合せとして寄り道探索を実現できないため，不可能である．

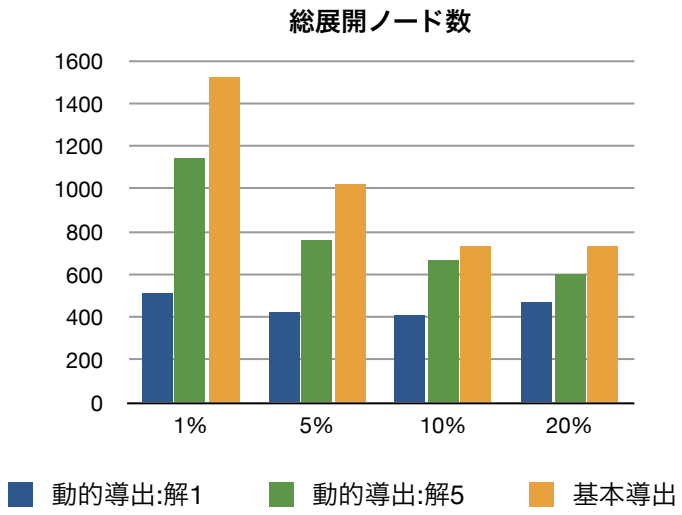


図 5-16: 基本導出法と動的導出法の比較（総展開ノード数）

5.7.2 時間制約つき寄り道探索の問合せ最適化

本節では時間制約つき寄り道探索の処理が，情報源の能力や分散形態によって，どのように実行されるかを示す．ここで注意すべき点として，基本導出法は，時間制約なしの寄り道探索の解を求めてから，時間制約をチェックするという 2 段階処理であるために，単一データベースの場合は不利であったものの，分散データベース構成では寄り道探索を情報源側に実行させられる場合があるため，動的導出法に比べて有利になることが期待されることである．以下に最適化で考慮される 3 つの観点を示す．

情報源に対する能力の仮定　前節で述べたような 3 つのレベルが想定される．

時間制約寄り道探索の処理方式の選択　データ統合側で基本導出法，または動的導出法を選択できる．

データの分散パターン　データの分散には図 5-18 に示すような 3 パターンがある．パターン 1 は集中である．パターン 2 は分散 1 と呼ぶが，店へ至るまでの交通情報と，店の詳細情報が情報源が分かれているものである（駅と店の繋がりは交通情報側に持つ）．パターン 3 は分散 2 と呼ぶが，店を含まない交通情報と，店情報に情報源が分かれているものである（駅と店の繋がりは店情報側に持つ）．

この 3 つの観点に対し，本手法を適用するときに，まずデータの分散パターンに対しては，既存技術と同じ，可能な限り 1 つの情報源から情報を取得しよ

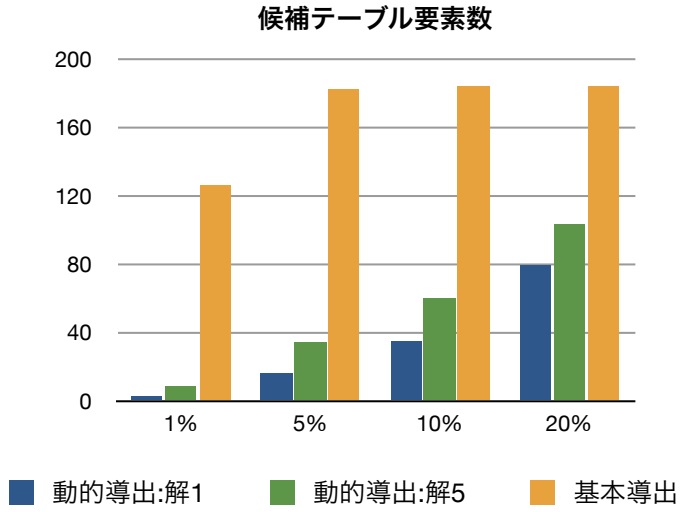


図 5-17: 基本導出法と動的導出法の比較（候補テーブル要素数）

うとする戦略に従うとする．情報源の能力と処理方式の選択については，前章で述べた階層管理を利用したプッシュダウン戦略に従う．

この 3 つの観点を組み合わせ，情報源側とデータ統合検索側の処理分担が決定される．表 5-3 は，その情報源側の処理を示している．残りの処理がデータ統合検索側で実行されることとなる．可能な限りレベルの高い処理を情報源側で実行する戦略に従うため，情報源能力がレベル 2 で，分散 1 で，基本導出法の場合は，寄り道探索処理はプッシュダウンすることが可能である．基本導出法と動的導出法のプッシュダウンの実現のイメージを図 5-19，図 5-20 にそれぞれ示す．

5.7.3 実験と評価

プッシュダウン効果を評価する実験を行った．首都圏鉄道網をグラフデータベース化し，全ノード約 1%のサービスノードを作成した（ノード・エッジとも約 4000 件）．データとしては小規模だが，グラフの局所的性質が同じであり，プッシュダウン効果の一次評価としては十分である．実験環境の構成を図 5-21 に示す．グラフデータベースには Neo4j[A75] を用いている．情報源は先に述べた分散パターンによって，3 組（集中・分散 1・分散 2）のデータベース環境を構築した．グラフデータベースの基本能力は，Neo4j の機能をそのまま用い，寄り道探索，時間制約つき寄り道探索等は Neo4j のユーザ定義関数の組み込み機能を用いて実装し，REST API 経由でアクセスしている．

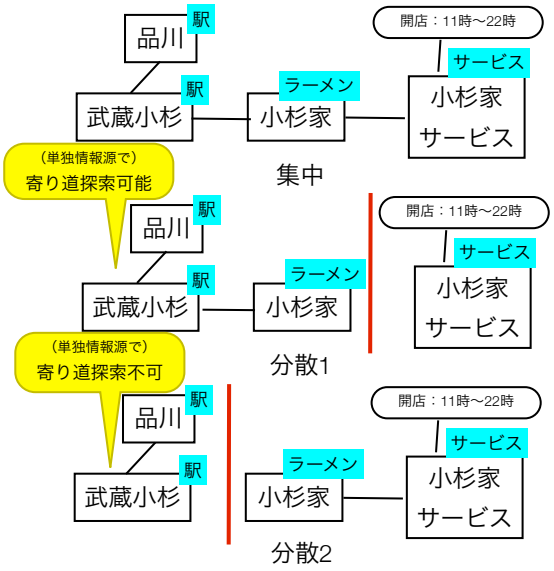


図 5-18: データの分散パターン

行き・帰りともに 40 分以内という条件で，時間制約つき寄り道探索を，前節で述べた観点の組み合わせ毎に測定した．評価結果を表 5-4 に示す（単位：秒）．

表内の枠で囲んだ部分がグラフ探索能力のプッシュダウンが適用されている範囲である．基本導出法においては，プッシュダウンを行える範囲が広く，動的導出法に比べて実用的な性能の領域が広いことがわかる．情報源能力がレベル 1 であったり，基本能力のみで探索を行う場合，ノードやエッジを取得するたびに，ネットワークアクセスが発生する．ノード・エッジの取得は，グラフ探索の過程において非常に多いため，性能は劣悪である．プッシュダウンの適用がある場合，ない場合という条件が合致しているときに，基本導出法と動的導出法を比較すると，動的導出法の結果が優れている．これは文献 [B79] における単一データベースにおける検証と合致している．

この評価結果は，プッシュダウンを利用した本方式の最適化戦略が有効であることを示している，また，時間制約つき寄り道探索を異種分散データベース環境で実現する場合は，基本導出法の選択が有効であることも示唆している．

5.8 まとめ

本章では，時間制約つき寄り道探索の考え方を提案し，その実現方法として基本導出法と動的導出法を明らかにした．提案方式を単一のグラフデータベースを利用して実装，評価し，動的導出法がサービス密度が低い場合に特

表 5-3: 最適化観点の組み合わせと情報源側の処理

導出法	基本導出法			動的導出法		
観点1→(能力)	L1	L2	L3	L1	L2	L3
集中	基本	寄り道	時間寄り道	基本	基本	時間寄り道
分散1	基本	寄り道	寄り道	基本	基本	基本
分散2	基本	基本	基本	基本	基本	基本

表 5-4: プッシュダウン効果検証実験の結果

単位：秒

導出法	基本導出法			動的導出法		
観点1→(能力)	L1	L2	L3	L1	L2	L3
集中	257.4	2.2	2.1	180.6	180.6	0.5
分散1	254.8	2.3	2.3	181.6	181.6	181.6
分散2	198.6	198.6	198.6	124.4	124.4	124.4

に有効であることを示した．さらに異種分散データベース環境で，時間制約つき寄り道探索を実現するために，本研究で提案した制約つきグラフ探索のための異種データベース統合技術を適用し，分散の場合は処理の分解が可能な基本導出法が性能の実用的な範囲が広いことを示した．

時間制約つき寄り道探索に関する今後の課題を述べる．異種分散データベース環境における場合の課題は，本研究全体のまとめで述べることとする．まず，課題として，複数の POI への拡張がある．ダイクストラ法を基本とする本手法では複数 POI へ拡張する場合，中間解における組み合わせが巨大になる問題が発生する．よって，本手法のように最適解を求めることが難しいため，非最適解を精度，効率良く求める手法が必要になる．また，本技術を適用する観点では，本章では鉄道網をベースとした典型的な利用シーンを想定して評価したが，本手法は汎用的であるため，より広い利用シーンで利用可能である．今後，twitter 等を利用して積極的な時間限定セールのマーケティングを行うとすると，本技術との組み合わせで新しいサービスが実現できる

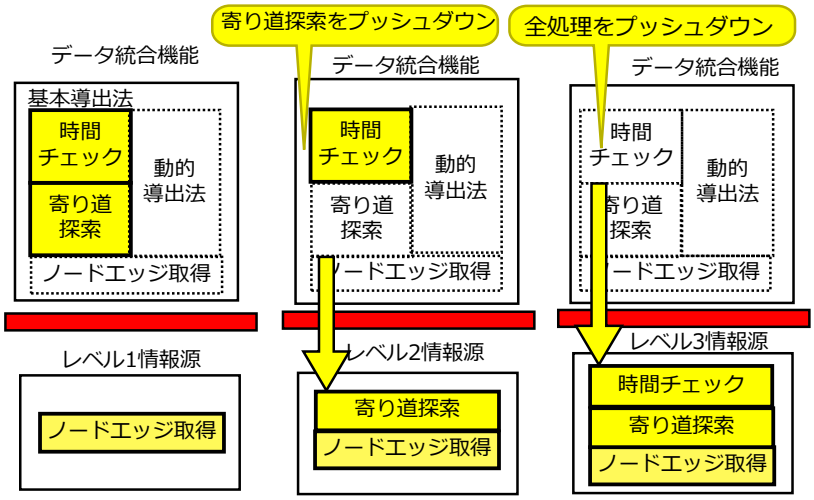


図 5-19: 情報源側への処理プッシュダウン：基本導出法

可能性もある．今後，サービスイメージや利用シーンをさらに検討し，その要求条件に沿ったモデルによる評価が課題となる．

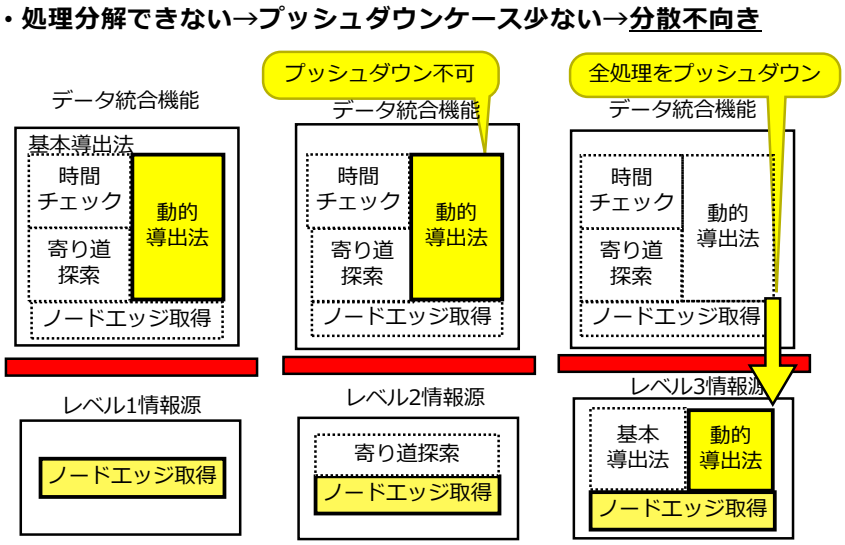


図 5-20: 情報源側への処理プッシュダウン：動的導出法

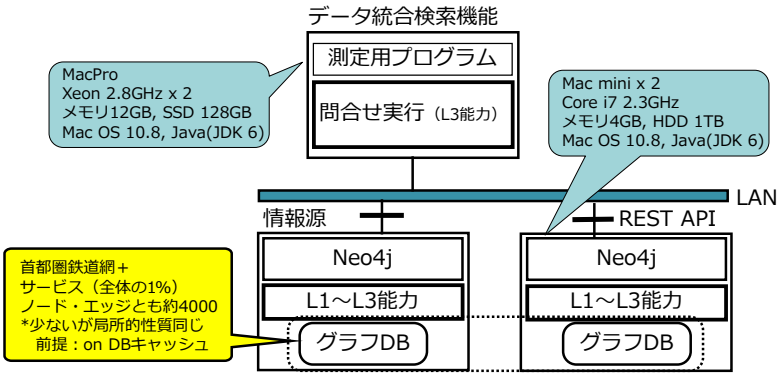


図 5-21: 実験環境の構成

第6章　考察

本章では，提案手法の特徴と適用範囲について考察する．

6.1　提案手法における特徴の評価

本論文における提案手法は 2 つの手法（技術）からなっていた．

- 手法 1：概念グラフを利用したスキーマ統合技術
- 手法 2：動的に異種性解消するデータ統合検索

手法 1 と手法 2 には，手法 1 で得られた対応関係メタデータを，手法 2 で活用するという関係があり，お互いに相補的である．1・2 を合わせた手法は特に以下の 3 点で特徴づけられる．それぞれの特徴について考察する．

- メタデータへのグラフモデル利用（概念グラフ）（主に手法 1）
- 動的特性（主に手法 2）
- プッシュダウン最適化（主に手法 2）

6.1.1　メタデータへのグラフモデル利用

提案手法では，スキーマ統合のために，概念グラフを利用した．概念グラフを利用することによって，構造衝突の回避が可能となり，過去のスキーマ統合手法に比較して，シンプルでツール化しやすい強力な手法を得ることができた，また，データ項目の標準化を用いた分解は，概念グラフとの相性がよく，概念のマッチングを増すために有効に働いた．

Sowa 氏による概念グラフは，現在は必ずしも一般的とはいえないが，ポイントは型構成要素の少ないモデルを採用することであり，RDF 等のモデルにも容易に展開することが可能である．

グラフデータベースへの適用という観点では，提案手法では，一度 ER モデルにマッピングしてから，概念グラフに変換している．その点では，メタデータのグラフモデルは，直接的にはグラフデータベースを扱うことに対して貢献してはいない．しかし，グラフデータベースのモデルやスキーマは今

後整備されていくことが予想され，その流れの中で，メタデータとしてのグラフを有効に利用できるようになると考えている．

本研究における評価では，通信ネットワークに関するデータベースを利用した．この評価自体は，グラフデータベースそのものではないものの，管理している情報は，グラフそのものであり，現在ならばグラフデータベースを利用する可能性が高い情報である．よって，その評価はグラフデータベースに適用した場合も有効であると考えている．ただし，実際の複数のグラフデータベースのスキーマの分析や統合は今後の課題ではある．

6.1.2 動的特性

動的特性とは，ユーザが指定する問い合わせに対して，実際に探索を実行する情報源を問い合わせ時に決定することを示している．この動的特性は，情報源の追加・変更に対して有効であり，特にグラフ情報利用サービスのような新しい領域では有効であると考えている．

動的特性の有効性については，本論文では直接的な評価は行なっていないが，実用化したシステム（MediPresto/M）の適用事例においては，完全に動的なアプリケーションは作られなかったものの，情報源の追加の容易性等のメリットが得られている．しかし，この動的特性の有効性の数値化は，検証がかなり難しく，今後の課題ではある．

6.1.3 プッシュダウン最適化

本論文のプッシュダウン最適化は，仮に情報源側で高度な能力を持っていない場合でも，基本能力が公開されていれば，基本能力を利用して，データ統合側で高度な能力を使うことによって，論理的にはすべてのグラフ処理を行うことが可能になるという点では，十分一般的である．グラフ探索はその原理上，基本能力の組み合わせで実現できるからである．ただし，実用的な観点から考えると，実験結果に示したように，基本能力のみでは実用的な性能を得ることは難しく，セキュリティの観点から基本能力が公開されないということもあるだろう．しかし，本論文は，今後の分散グラフデータベースの最適化のための重要な最初の一歩となっている．

また，このような最適化の観点については，時間制約つき寄り道探索に関する実験結果が示すように，最適化が有効である可能性は高い．ただし，部分グラフをまるごととるような単純な処理の組み合わせが有効な場合も，グラフのデータ分布によっては考えられるため，今後のグラフ処理の汎用モデル化の中でさらに検討していく必要がある．

6.2 提案手法の適用範囲と限界

本手法は，グラフ探索処理を仮想的な表としてモデル化しており，Web インターフェースとして公開されていることを前提としている．この仮定は一般性を有していると言える．本論文における寄り道探索の経由点は一箇所という想定であったが，それが複数箇所になった場合でも表モデルでモデル化は可能だから，本手法の適用は可能である（ただし，寄り道探索のアルゴリズムは拡張の必要がある）．グラフ探索は，そのパラメータ指定はノードやそのプロパティであることが多いため，適用範囲は広い．ただし，本手法の前提は，情報源の仮想化を関係モデルとすることであった．よって，実用上の課題としては，寄り道探索の返却結果は，経路情報を JSON 等の文字列で返却されることが想定されており，この文字列を分解し，必要な情報を取り出す処理は利用者（データ統合検索の開発者）側で実施する必要がある．経路情報の表現形式変換関数も利用者責任で作成する必要がある．また，グラフ類似検索・軌跡検索のようにグラフをパラメータとして渡す場合，それらを文字列等に変換して情報源側に渡せば適用は可能であるが，項目の対応を基礎としている本手法のメリットは必ずしも生かせない．

また，本手法は，情報源を跨るグラフ「探索」は対象外であり，情報源を跨るデータ統合処理は，関係演算である結合と和に限られている．例えば，鉄道網の最短経路探索が各鉄道会社毎に構築されている場合，両社の路線を辿って検索する結果を求めることができず，最低レベルのグラフ探索能力（ノード・エッジ取得のみ）を用いて探索を実施するために，性能は劣化する．このような分散的なグラフ探索を効率的に扱う問題は，個別の探索毎に有効な手法も異なるため，一般的な異種分散データベースのフレームワーク内で整理するのは，今後の課題である．

6.3 提案手法の応用

本論文では提案手法を，時間制約寄り道探索へ適用した．しかし，これまでの考察でもわかるように，本手法の応用範囲は広く，寄り道探索に限らず，他の旅行計画問い合わせへの応用も可能である．また，最初の例 1 で紹介した SNS の制約探索も応用範囲に含まれる．

本手法により，単一情報源では実現できないグラフデータ統合検索を容易に実現することできる．初期の代表的な応用としては，グラフデータベースに対して，他の情報源が持つ情報を付加するような探索があげられるだろう．例えば，ノードにレストランの名前を持つときに，他のデータベースにレストランを分類する階層情報やレビュー情報がある場合に，それらを結合したグラフ探索が可能となる．

第7章　結論

7.1　本研究の到達点

ノードまたはエッジのプロパティに対して制約条件を付加し，グラフ探索を実施する制約つきグラフ探索を異種分散データベース環境において実現することを研究の目標とした．その達成のためには，課題 1：スキーマ間の異種性解消と統合，課題 2：情報源のばらつきに対応した問い合わせ処理，の 2 つの課題を解決する必要があった．

課題 1 を解決するために，概念グラフを利用したスキーマ統合技術（手法 1）を提案し，課題 2 を解決するために，動的に異種性解消するデータ統合検索（手法 2）を提案した．2 つの手法は，手法 1 の過程で概念・データ項目・表現形式の間の関係を表すメタデータが得られ，それを利用し検索することによって，手法 2 の動的なデータ統合検索を実現するという関係にある．

手法 1 においては，ER モデル・グラフデータベースモデルを適切に概念グラフに変換することができ，その情報の突き合わせ手法も確立され，スキーマ統合ツールを利用することによって稼働を約 1/3 に削減できる見通しを得た．

手法 2 においては，データの所在を意識させない，問合せインターフェースを利用し，検索時にメタデータを探索し，統合対象を決定するアプローチをとった．本アプローチは，スキーマの変更に強い点が特長であり，進化しつつあるグラフデータベースのサービス変更に対応するために有効である．データ統合は，関係モデルに基づいて実現することができ，制約つきグラフ探索機能を，Web 情報源機能の拡張により仮想的な表としてモデル化し統合可能とした．さらに制約つきグラフ探索処理を階層的に分解・管理することにより，情報源の能力に応じた，情報源側へのプッシュダウンを可能とした，

提案手法を，制約つきグラフ探索の事例である時間制約つき寄り道探索へ適用した．まず，時間制約つき寄り道探索を定義し，定式化した．そして，時間制約を利用した探索の枝刈りを行い，探索空間を削減することが可能となる動的導出法を提案し，単一データベースの場合において性能が優れることを示した．さらに，複数データベースの場合における評価を実施し，3 つの観点（導出法・能力レベル・分散）により本手法を適用した問い合わせ分解を示し，処理が階層化されており，プッシュダウンが可能となる基本導出法が性能が実用的になる範囲が広いことを示した．

7.2 今後の課題

7.2.1 提案手法の適用検証

本研究では，時間制約つき寄り道探索という制約つきグラフ探索の一例を用いて，提案手法の有効性を示したが，さらなる適用検証を行うことにより，提案手法の有効性の高い適用分野を見極められると考えられる．時間制約つき寄り道探索は，自明でもなく，また過度に複雑でもない制約つきグラフ探索であると言えるが，より単純な制約つきグラフ探索である top-k 最短経路探索とプロパティ条件を組み合わせた検索や，より複雑なグラフ探索である POI を複数化した寄り道探索への適用が今後の適用検証の候補となる．性能・統合構築の容易性等の観点で，理論的または実用的な評価・分析を行い，関係モデルに基づいた本手法がどの程度の範囲で有効であるかを示せればよい．

7.2.2 本手法の拡張

本手法は関係モデルに基づいていたために，問合せがグラフの構造を反映することができないことと，複数の情報源の統合処理が結合と和という関係モデル演算のみであるという大きな二つの制約があった．その二つの制約を緩和し，汎用的なグラフデータ処理における異種分散データベース技術を今後確立していく必要がある．

まず，問合せ拡張の候補としては，グラフ同士を比較するパターンマッチングや，RDF グラフを処理する SPARQL 言語の処理（主に知識を表現する三つ組であるトリプルのマッチング処理）の重要性が現在高いと考えられる．分散グラフ処理への拡張は，その探索処理の特長によって選択すべき手法が異なる可能性が高く，本研究における提案手法のような一般化は難しい．どちらの拡張も，根本的な処理モデルの見直しが必要であり，困難な課題である．汎用グラフ処理を一般化・モデル化し，演算・操作体系や，処理の基本操作への分解方法，最適化に利用するための操作の交換ルール等を新たなモデルで確立する必要である．

参考文献

各章の参考文献

— 第 **1**，**2** 章参考文献 —

[A1] Shekhar, S. and Chawla, S. : Spatial Databases: A Tour, Prentice Hall (2003).

[A2] Tansel, A. U., Clifford, J. , Gadia, S.: Temporal Databases: Theory, Design, and Implementation, Benjamin-Cummings(1993).

[A3] Cheng, J., Ke, Y., NG, W.: Efficient Query Processing on Graph Databases, ACM Transactions on Database Systems, Vol. 34, No. 1, pp. 1-48 (2009).

[A4] Neo4j WEB Page, http://neo4j.org

[A5] Sheth, A.P. and Larson, J.A. : Federated database systems for managing distributed, heterogeneous, and autonomous databases. ACM Computing Survey, Vol. 22, No.3 , pp.183-236 (1990)

[A6] Batini, C. , Lenzarini, M. , Navathe, S.B. : A Comparative analysis of methodologies for database schema integration, ACM Computing Survey, Vol.18, No. 4, pp.323-364 (1986).

[A7] Kim, W. and Seo, J. : Classifying Schematic and Data Heterogeneity in Multidatabase Systems, Computer, Vol.24, No.12, pp.12-18 (1993).

[A8] Hammer, J. , Garcia-Molina, H. , Ireland, K. , Papakonstantinou, Y. , Ullman, J. , and Widom, J. : Information Translation, Mediation, and Mosaic-Based Browsing in the TSIMMIS System, Proceedings of the ACM SIGMOD International Conference on Management of Data, pp.483 (1995).

[A9] Levy, A.Y., Rajaraman, A. Ordille, J.J. : Querying Heterogeneous Information Sources Using Source Descriptions, Proceedings of the 22nd VLDB Conference (1996).

[A10] Data Integration:Underlying Problems and Research Approaches, ETH Group (2010).

[A11] Lenzarini, M. : Data Integration: A Theoretical Perspective, In Proceedings of the 21st ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS '02) (2002).

[A12] Zaman, M. : Information Integration for Heterogeneous Data Sources, IOSR Journal of Engineering Vol. 2(4) pp.640-643 (2012)

[A13] Madhavan, J., Bernstein, P. A. , and Rahm, E. : Generic Schema Matching with Cupid. Proc. VLDB, pp.49-58, (2001).

[A14] Chen, Z., Shen, H.T., Zhou, X., and Zheng, Y.: Xing XieSearching Trajectories by Locations ? An Efficiency Study, In ACM SIGMOD International Conference on Management of Data (SIGMOD 2010), Indianapolis, Indiana, USA.

[A15] Goldberg, A.V., Harrelson, C.: Computing the shortest path: A search meets graph theory , In ACM-SIAM Symposium on Discrete Algorithms - SODA , pp. 156-165, 2005

[A16] Gutman, R.J. , Reach-Based Routing: A New Approach to Shortest Path Algorithms Optimized for Road Networks, in Proc. ALENEX/ANALC, pp.100-111, 2004.

[A17] Wagner, D., Willhalm, T.: Geometric Speed-Up Techniques for Finding Shortest Paths in Large Sparse Graphs, In Proc. 11th Annual European Symposium, Budapest, Hungary, September pp.16-19 (2003).

[A18] Levy, A.Y., Rajaraman, A. Ordille, J.J. : Querying Heterogeneous Information Sources Using Source Descriptions, Proceedings of the 22nd VLDB Conference (1996).

[A19] Halevy, A. , Rajaraman, A. Ordille, J.J. : Data integration: the teenage years. In Proceedings of the 32nd international conference on Very large data bases, pp.9-16 (2006).

[A20] Madhavan, J., Bernstein, P. A. , and Rahm, E. : Generic Schema Matching with Cupid. Proc. VLDB, pp.49-58, (2001).

[A21] Bernstein, P. A. , Madhavan, J. , Rahm, E. : Generic Schema Matching, Ten Years Later. PVLDB 4(11): pp.695-701 (2011)

— 第 **3** 章参考文献 —

[A22] Batini, C. , Lenzarini, M. , Navathe, S.B. : A Comparative analysis of methodologies for database schema integration, ACM Computing Survey, Vol.18, No. 4, pp.323-364 (1986).

[A23] 鈴木源吾，山室雅司，中渡瀬秀一：スキーマ統合におけるスキーマ要素間の類似性発見手法，電子情報通信学会 1994 年秋季全国大会，D-62 (1994).

[A24] Navathe, S.B. , Gadgil, S.G. : A methodology for view integration in logical database design, 8th Int.Conf. on VLDB. pp.142-155 (1982).

[A25] Hayne, S. , Ram, S. : Multi-user view integration system(MUVIS): An expert system for view integration, the 6th IEEE International Conference on Data Engineering Proceedings, pp.402-409 (1990) .

[A26] Batini, C. , Lenzarini, M. : A methodology for data schema integration in the entity relationship model, IEEE Transaction on Software Engineering, Vol.SE-10, No.6, pp.650-664, (1984).

[A27] Spaccapietra, S. , Parent, C. : View integration: a step forward in solving structural conflicts, IEEE Transaction on knowledge and data engineering, Vol.6, No.2, pp.258-274 (1994).

[A28] 関根，川下，町原，中川：体系的な DB 構築のための用語辞書を用いたデータ標準化手法，情報処理学会論文誌，Vol.34, No.3, pp.457-467 (1993).

[A29] Durell, R.W. : Data Administration, McGraw-Hill (1985).

[A30] Sowa, J.F. : Conceptual structures: information processing in mind and machine, Addison-Wesley (1984).

[A31] Creasy, P. , Ellis, G. : A conceptual graphs approach to conceptual schema integration, Lecture Notes in AI 699 Proceedings of International Conference on Conceptual Structures, Springer Verlag (1993) .

[A32] Kitagawa, T. , Kiyoki, Y. : A mathematical model of meaning and its application to multi database systems, RIDE-IMS'93 Proceedings, pp.130-135. (1993).

— 第 4 章参考文献 —

[A33] Baker, R.H. : EXTRANETS, McGraw-Hill, New York (1997).

[A34] 財団法人データベース振興センター編：データベース白書 1998，財団法人データベース振興センター，東京 (1998).

[A35] Navathe, S. and Savasere, A. : A Schema Integration Facility Using Object-Oriented Data Model in [A36], pp.105-128.

[A36] Bukhres, O. A. and Elmagarmid, A. K. ed. : Object-Oriented Multidatabases Systems: A Solution for Advanced Applications. Prentice-Hall, New Jersey (1996).

[A37] Bouguettaya, A., Benatallah, B. and Elmagarmid, A. : An Overview of Multidatabase Systems: Past and Present, in [A38], pp.1-32.

[A38] Elmagarmid, A., Rusinkiewicz, M. and Sheth, A. ed. : Management of Heterogeneous and Autonomous Database Systems. Morgan Kaufman Publishers, San Grancisco, CA (1998).

[A39] Ram, S. and Ramesh, V. : Schema Integration: Past, Present and Future. in [A38], pp.119-156.

[A40] Gotthard, W., Lockemann, P. and Neufeld, A. : System Guided View Integration for Object-Oriented Databases. IEEE Trans. on Knowledge and Data Engineeringm Vol.4, No.1, pp.1-22(1992).

[A41] Bouzeghoub, M. and Comyn-Wattiau, I. : View Integration by Semantic Unification and Transformation by Semantic Unification and Transformation of Data Structures. Proc. Int. Conf. on Entity-Relationship Approach(ER'90), pp.413-430 (1990).

[A42] Kim, W. and Seo, J. : Classifying Schematic and Data Heterogeneity in Multidatabase Systems, Computer, Vol.24, No.12, pp.12-18 (1993).

[A43] 池田哲夫，伊土誠一，石垣昭一郎，村田達彦：データ流通プラットフォームシステム：DB-STREAM，情報処理学会論文誌，Vol.38, No.12, pp.2552-2565 (1997).

[A44] Ullman, J. D. : Principles of Database and Knowledge-Base Systems: The New Technologies. Vol.2, W H Freeman & Co. (1989).

[A45] NTT 情報通信研究所（池田哲夫，川下満，坂田哲夫，関根純，中川優，村田達彦）：データベース概念設計第 2 版，阿部出版，東京 (1996).

[A46] NTT 情報通信研究所（大久保成隆，川下満，関根純，中川優，町原宏毅，村田達彦)：データベース論理設計第 2 版，阿部出版，東京 (1996).

[A47] 関根純，川下満，町原宏毅，中川優：体系的な DB 構築のための用語辞書を用いたデータ標準化手法，情報処理学会論文誌, Vol.30, No.3, pp.457-467 (1993).

[A48] Durell, R.W. : データ資源管理，日経マグロウヒル (1987).

[A49] Darwen, W. and Datte, C. J. : A Guide to the SQL Standard : A User's Guide to the Standard Database Language SQL, Addison-Wesley (1997).

[A50] Zhao, J. L., Segev, A. and Chatterjee, A : A Universal Relation Approach to Federated Database Management, International Conference on Data Engineering (ICDE), pp.261-270 (1995).

[A51] Reck, C. and Hillebrand, G. : Implementing a Universal Relation Interface Using Access Scripts with Binding Patterns, Technical Report 1996-40, Universitat Karlsruhe (1996).

[A52] Reck, C. and Konig-Ries, B. : An Architecture for Transparent Access to Semantically Heterogeneous Information Sources. Proc. 1st Cooperative Information Agents (CIA) 1997, Springer-Verlag, Berlin, pp.260-271 (1997)

[A53] Yu, C. T., Sun, W., Dao, S. and Keirsey, D.: Determining Relationships among Attributes for Interoperability of Multi-Database Systems, International Workshop on Interoperability in Multidatabase Systems (RIDE-IMS91), Kyoto, pp.251-257 (1991)

[A54] Bright, M. and Hurson, A. : Linguistic Support for Semantic Identification and Interpretation in Multidatabases. International Workshop on Interoperability in Multidatabase Systems (RIDE-IMS91), Kyoto, pp.306-313 (1991)

[A55] Collet, C., Huhns, M. N. and Shen, W. : Resource Integration Using a Large Knowledge Base in Carnot. Computer, Vol.24, No.12, pp.55-62 (1991).

[A56] Batini, C. , Lenzarini, M. , Navathe, S.B. : A Comparative analysis of methodologies for database schema integration, ACM Computing Survey, Vol.18, No. 4, pp.323-364 (1986).

[A57] Sheth, A.P. and Larson, J.A. : Federated database systems for managing distributed, heterogeneous, and autonomous databases. ACM Computing Survey, Vol. 22, No.3 , pp.183-236 (1990)

[A58] Levy, A.Y., Rajaraman, A. Ordille, J.J. : Querying Heterogeneous Information Sources Using Source Descriptions, Proceedings of the 22nd VLDB Conference (1996).

[A59] Hammer, J. , Garcia-Molina, H. , Ireland, K. , Papakonstantinou, Y. , Ullman, J. , and Widom, J. : Information Translation, Mediation, and Mosaic-Based Browsing in the TSIMMIS System, Proceedings of the ACM SIGMOD International Conference on Management of Data, pp.483 (1995).

— 第 **5** 章参考文献 —

[A60] Dijkstra, E. W. : A note on two problems in connexion with graphs, Numerische Mathematik Vol.1, pp. 269-271 (1959).

[A61] Hart, P. E. , Nilsson, N. J. , Raphael, B. : A Formal Basis for the Heuristic Determination of Minimum Cost Paths, EEE Transactions on Systems Science and Cybernetics Vol.SSC4, No.2, pp. 100-107 (1968).

[A62] 五十嵐健夫：データ構造とアルゴリズム，数理工学社 (2007).

[A63] 駅前探検倶楽部 http://ekitan.com/

[A64] 食べログ http://tabelog.com/

[A65] 大沢裕，藤野和久：前処理を必要としない道路ネットワーク上での最短寄り道経路探索アルゴリズム，電子情報通信学会論文誌 D， Vol.J93-D, No.3, pp. 203–210 (2010).

[A66] Sharifzadeh, M., Kolahdouzan, M., and Shahabi, C.: The optimal sequenced route query, *VLDB J.* , Vol.17, pp.765–787 (2008).

[A67] Li, F., Cheng, D., Hadjieleftheriou, M., Kollios, G., and Teng, S.-H.: On trip planning queries in spatial databases, *SSTD 2005 Proceedings* , pp. 273–290 (2005).

[A68] Papadias, D., Zhang, J., Mamoulis, N., and Tao, Y.: Query processing in spatial network databases, *29th VLDB Proceedings* , pp. 790–801 (2003).

[A69] 蒲原智也，上島紳一：道路網応用のための空間索引木の提案と最短経路探査への応用，情報処理学会論文誌. データベース Vol.2, No.2, pp. 10-28 (2009).

[A70] Tansel, A. U., Clifford, J. , Gadia, S.: Temporal Databases: Theory, Design, and Implementation, Benjamin-Cummings(1993).

[A71] Norvag, K.: Temporal Query Operators in XML Databases, *ACM Symposium on Applied Computing 2002 Proceedings* , pp. 402–406 (2002).

[A72] Kleinberg, J. and Tardos, E.:アルゴリズムデザイン，共立出版 (2008).

[A73] 松田三恵子，杉山博史，土井美和子：歩行者の経路への嗜好を反映した経路生成，電子情報通信学会論文誌 A， Vol.J87-A, No.1, pp. 132–139 (2004).

[A74] Cheng, J., Ke, Y., NG, W.: Efficient Query Processing on Graph Databases, ACM Transactions on Database Systems, Vol. 34, No. 1, pp. 1-48 (2009).

[A75] Neo4j WEB Page, http://neo4j.org

主要発表論文等一覧

— 論文誌 —

[B76] 山室雅司，鈴木源吾：データ標準化と概念グラフへの変換を利用したスキーマ統合支援法，電子情報通信学会論文誌 D-1 Vol.J79-D-I No.11 pp.966-974 (1996).

[B77] 池田哲夫，鈴木源吾，町原宏毅，安田浩：連邦データベースシステムにおけるスキーマ構築の一方式，情報処理学会論文誌 Vol.40 No.SIG8(TOD 4) pp.29-40 (1999).

[B78] 林孝志，小西一也，堀口恭太郎，綱川光明，鈴木源吾，芳西崇：異種情報源統合のための XML 問い合わせ最適化と情報源問合せ能力管理，情報処理学会論文誌 Vol.44 No.SIG12(TOD 19) pp.1-10 (2003).

[B79] 鈴木源吾，榎本俊文，小林伸幸，山室雅司，鬼塚真：時間制約をもつ寄り道経路探索システムの実現と評価，情報処理学会論文誌 Vol.53 No.2 pp.857-867(2012).

— 国際会議 —

[B80] Suzuki, G. , Yamamuro, M. : Schema Integration Methodology Including Structural Conflict Resolution and Checking Conceptual Similarity - Conceptual Graphs Approach - , International Workshop on Database Reengineering and Interoperability, pp.229-242 (1995).

[B81] Suzuki, G. , IIzuka, Y. , Kasuga, S. : Integration of Keyword Bases Source Search and Structure Bases Information Retrieval, 7th International CODATA Conference, pp.149-158 (2000).

[B82] Honishi, T. , Suzuki, G. , Kobayashi, N. , Konishi, K. : A Mediation System Based on Universal Relation Modeling, 20th International Conference on Conceptual Modeling Proceedings (ER2001), SE3, pp.1-4 (2001).

[B83] Hayashi, T. , Suzuki, G. , Iizuka, Y. , Konishi, K. Honishi, T. : Distributed Multimedia Information Retrieval that Accepts Arbitrary Media Key, 7th Workshop on Multimedia Information Systems, pp.153-162 (2001).

— シンポジウム（査読あり） —

[B84] 鈴木源吾，山室雅司：構造衝突の解消と概念類似性の判定を両立したスキーマ統合，情報学シンポジウム (1995).

— 特許 —

[B85] 鈴木源吾：データベース異種性解消検索装置，特許 3438805 号.

[B86] 町原宏毅，鈴木源吾，岡田英二，加納直哉：情報検索方法および情報検索システム，特許 3786233 号.

[B87] 鈴木源吾，網川光明，飯塚裕一，瀬尾 紳一郎：異種情報源問い合わせ変換方法及び装置，特許 3671765 号.

[B88] 小西一也，鈴木源吾，飯塚裕一:統合検索方法及び装置，特許 3558032 号.

— 研究会資料等（査読なし） —

[B89] 鈴木源吾，鬼塚真，榎本俊文，小林伸幸：制約つきグラフ探索を実現する異種データベース統合技術，情報処理学会研究報告 2013-DPS-154 (2013).

[B90] 鈴木源吾，山室雅司，中渡瀬秀一：スキーマ統合におけるスキーマ要素間の類似性発見手法，電子情報通信学会 1994 年秋季全国大会，D-62 (1994).

[B91] 鈴木源吾，山室雅司：データベース概念スキーマ統合支援ツールの検討，電子情報通信学会 1994 年ソサイエティ大会，D-31 (1995).

[B92] 鈴木源吾，町原宏毅，川下満：Fragment View- マルチデータベースにおける Global View を使わない異種性解消方式，信学技報 DE96-78 (1997).

[B93] 鈴木源吾，町原宏毅：データベースの値の範囲の管理法とその普遍関係ユーザインターフェースへの応用，信学技報 DE97-2 (1997).

[B94] 鈴木源吾，小西一也，林孝志，小林伸幸，芳西崇：XML に基づく異種情報源メディエーションシステム：MediPresto/XM，情報処理学会研究報告 2001-DBS-125-60 (2001).

謝辞

本論文をまとめるにあたって多くの方にご助言をいただきました．ここに各位のお名前を記して感謝申し上げます．

まず，社会知能情報学専攻 鬼塚真 教授には，当該分野の調査から，研究の計画，研究内容に関する議論，結果のまとめ，プレゼンテーション等全てに渡ってご指導をいただきました．審査員の社会知能情報学専攻 大須賀昭彦 教授，田中健次 教授，情報システム基盤学専攻 大森匡 教授，情報メディアシステム学専攻 田野俊一 教授には，報告会や研究審査において貴重なコメントをいただきました．また，太田敏澄 先生には，ゼミでの発表の指導や議論，論文作成に関して，深いご意見とご指導をいただきました．さらに，本研究についての助言や協力をいただいた鬼塚研究室と太田研究室のメンバーの皆様に感謝いたします．

本研究は，NTT ソフトウェアイノベーションセンタ，及び NTT サイバースペース研究所で行った研究を基としており，研究を実施していた当時に，ご指導とご協力をいただきました，芳西崇氏，町原宏毅氏，山室雅司氏，池田哲夫氏，小林伸幸氏，星野隆氏，綱川光明氏，飯塚裕一氏，春日史朗氏，小西一也氏，林孝志氏，榎本俊文氏に感謝いたします．

関連論文の印刷公表の方法及び時期

1. 鈴木源吾，山室雅司

   論文題目「Schema Integration Methodology Including Structural Conflict Resolution and Checking Conceptual Similarity - Conceptual Graphs Approach -」

   平成 7 年 3 月，International Workshop on Database Reengineering and Interoperability, pp.247-260（第 3 章内容に関連）

2. 鈴木源吾，飯塚裕一，春日史朗

   論文題目「Integration of Keyword Bases Source Search and Structure Bases Information Retrieval」

   平成 12 年 10 月，17th International CODATA Conference, pp.149-158（第 4 章内容に関連）

3. 芳西崇，鈴木源吾，小林伸幸，小西一也

   論文題目「A Mediation System Based on Universal Relation Modeling」

   平成 13 年 11 月，20th International Conference on Conceptual Modeling Proceedings (ER2001), SE3, pp.1-4（第 4 章内容に関連）

4. 鈴木源吾，榎本俊文，小林伸幸，山室雅司，鬼塚真

   論文題目「時間制約をもつ寄り道経路探索システムの実現と評価」

   平成 23 年 2 月，情報処理学会論文誌 Vol.53 No.2 pp.857-867（第 5 章内容に関連）

関連論文

SCHEMA INTEGRATION METHODOLOGY INCLUDING STRUCTURAL CONFLICT RESOLUTION AND CHECKING CONCEPTUAL SIMILARITY

Conceptual Graphs Approach

Gengo Suzuki and Masashi Yamamuro

NTT Information and Communication Systems Laboratories
1-2356, Take, Yokosuka-shi, Kanagawa, 238-03 Japan
e-mail: {gsuzuki, masashi}@ciladb.dq.isl.ntt.jp

Abstract: When integrating several conceptual database schemas, several kind of representation differences must be resolved. Such differences are called conflicts. The conflict on which the same concepts are represented by different modeling constructs in different schemas is called a "structural conflict". This paper points out problems in existing methodologies for schema integration which use the Entity-Relationship model. A method is proposed using conceptual graphs as a common data model, to adequately resolve these structural conflicts and to discover similar schema elements. Procedures for integration including the discovery of similar schema elements are clarified. And the methodologies for translating between schemas of ER model and conceptual graphs are shown.

1. INTRODUCTION

Recently a technology of schema integration has become paramount in the re-engineering of information and designing information for companies. In order to integrate schemas several representation differences must be resolved between schemas. Differences in which the same concepts are represented by different modeling constructs are called "structural conflicts". And when integrating schemas in different data models, they are translated to a common data model, which is a basis of discussions.

In existing researches concerning schema integration, the Entity-Relationship (ER) model [4] is used as a common data model. But the methodologies employing this ER model have some problems when resolving structural conflicts. So we propose a method that conceptual graphs are used as a common data model to integrate. This method is also effective for the following point. In schema integration, a technique for discovering similar schema elements is necessary, but in the ER method structural similarities are complex. We can simplify this complexity by using conceptual graphs.

Database Reengineering and Interoperability, Edited by To-yat Cheung et al.
Plenum Press, New York, 1996

In this paper the method of schema integration using conceptual graphs is described. In section 2, problems of ER approach and effectiveness of our method are shown, In section 3, outlines of definition of conceptual graphs and a procedure of schema integration using conceptual graphs are described. A problem to discover similar schema elements and a structural conflict resolution have never been discussed at the same time. Our procedure includes the both. In section 4 we describe how to translate schemas of ER model to conceptual graphs, and section 5 we show how to translate an integrated conceptual graph to a schema in ER model. In the last section application examples of our method are described.

2. SCHEMA INTEGRATION USING CONCEPTUAL GRAPHS

2.1. ER Approach Problems for Structural Conflict Resolution

A "structural conflict" is a situation in which the same concepts are represented by different modeling constructs [2, 13]. When schemas are integrated which have different data models, schemas are compared after they are translated into a common data model [11]. The ER model [4] has been used as the common data model for several projects up until now [1, 13]. Practical application of the ER model has uncovered several disadvantages concerning structural conflict resolutions.

The first is the selection of modeling constructs. There is freedom to select modeling constructs to represent one concept. The model designer is responsible for this selection, which may lead to structural conflicts. The result is that the ER model may be one cause of conflict.

The second is an inability to separate the domains and dependencies in attributes. In the ER model an entity type and a set of values of an attribute are not on the same level. When the ER model is used, it is said that an attribute corresponds to an entity type to refer to one of a structural conflict. But the meaning is not clear because an attribute in the ER model represents both a dependency between an entity type and a domain and a set of real world objects which the attribute represents. For example, the attribute "involving department" of an entity type "employee" represents two types of information, one of which is a set of department objects and the other is a set of dependency relationships between departments and employees.

The formal definition of the ER model includes the concept of domain, however this is only used for an attribute value data type, and does not correspond to an entity type in schema integration methodologies. In the papers by Spaccapietra and Larson [13, 10], "Real World State" is used for discussion, which is the real world domain corresponding to an attribute. This, however is unnatural because the "Real World State" is a concept outside the data model.

2.2. Similarity of Schema Elements

In order to support schema integration through computer automation, techniques for computing similarities between schema elements are necessary. These are discussed elsewhere [3, 9, 8, 14].

Similarities of schema elements are categorized into the following three types [14].

1. Similarity in Names

   Names of schema elements, such as the name of an attribute, are similar. For example, if two names of schema elements have many common character strings, they are

interpreted to be similar.

2. Similarity of Semantics

   Quantified semantics of schema elements are defined, and are used to compare elements for similarity. For example, in the paper by Kitagawa and Kiyoki [9] semantics of schema elements can be defined by a vector which represents contributions to several chosen keywords. And similarities are defined by using these vectors.

3. Similarity of Structures

   It is regarded as a similarity, if there are many similar concepts around two schema elements. For example, entity types which contain many common attributes are interpreted as similar elements.

However if some concept is modeled as an attribute in one schema and is modeled as an relationship type and an entity type in the another schema, a condition "There are many common attributes." does not contribute to the similarity of these schema elements which are really the same concepts. If we try to judge the similarity, the condition "There are many common attributes and relationships" must be considered. This conclusion would make judging unnecessarily complex, and as a result there is no such a consideration in existing theories. If a model which causes fewer structural conflicts is used, then the comparisons can be simplified.

2.3. Schema Integration Using Conceptual Graphs

If structural conflicts are to be eliminated, and the structural similarities of schema elements are to be simplified, a model that minimizes modeling constructs and clearly separates a domain and a dependency is needed.

The conceptual graph [12] is the model that satisfies these requirements. In this model, modeling constructs are a few, containing only "concept" and "relation". Both the Information Resource Dictionary Systems(IRDS) common data model, which is called a normative language, and the schema integration common data model have a common requirement which is a simple model. ANSI has proposed the conceptual graph as the standard candidate for the normative language and this has resulted in the conceptual graph's popularity as a simple model, such as that mentioned above.

The conceptual graph, however, is not adequate for modeling top-down database design, because due to its plain structure it does not allow the stepwise design from "entity types" to "attributes". Presently object oriented models and the ER model are popular as data models for database design. We propose a method for schema integration using the conceptual graph as a common data model. In this method, input schemas are represented by the ER model, then translated into input schemas for conceptual graphs, next integrated into one conceptual graphs, and finally the integrated conceptual graphs are translated to a schema of the ER model.

In existing researches, discovering similar schema elements and resolution of structural conflicts are not considered at the same time. Creasy and Ellis proposed a method using conceptual graphs, and emphasized that the method is effective in resolving data model conflicts [5]. However if a concept mapped from an attribute and a concept mapped from an entity type are considered as the same concepts, how to create an integrated ER schema is not clear. Hayne and Ram discuss schema integration by using an SDM [7], which makes no distinction between a value and an entity, however they do not address structural conflict resolutions [8].

This paper discusses a methodology for considering similarities of schema elements and structural conflict resolutions at the same time by using the conceptual graph as mediators.

3. CONCEPTUAL GRAPHS AND SCHEMA INTEGRATION USING CONCEPTUAL GRAPHS

3.1. Definition of Conceptual Graphs

The real world is modeled by concepts and relations in the conceptual graph. Usually a concept is represented as [type label: individual marker]. The type label means a type of a concept. The individual marker means an individual which the concept indicates. For example a concept [Girl: Judy] means an individual 'Judy' of a type 'Girl'. There is a special individual marker $\{*\}$ meaning all individuals which a type represents. This paper considers only concepts which represent all individuals of a type. So concepts is represented as only type label ($\{*\}$is omitted.). We can define generalization hierarchies using generalization relationships represented by "sub-type-label $\leq$ super-type-label". We can also define generalization relationships of relations, which are not in the original definitions of conceptual graphs (a similar notation is used.). Relations have their directions. Graphically concepts are represented by rectangles and relations are represented by ellipses which are connected to concepts.

3.2. Schema Integration Using Conceptual Graph

The following is the method for schema integration using conceptual graphs. Using this method steps can be clarified in which similarities of schema elements are computed. Detailed discussions of similarity computations are curtailed due to the length of this paper.

1. Translate input schemas to conceptual graphs

   In section 4 we will discuss the translation.

2. Clarify similarities between individual concepts

   Similarities between individual concepts mean naming and semantic similarities described in 2.1 and a specified portion of the constraint similarities such as a domain similarity, which can be defined by checking individual concepts.

3. Clarify similarities taking into consideration neighborhood properties between concepts

   This means that structural similarities which cannot be defined by only the concept informations which are views as separate. For example, a pair of concepts are considered similar when neighboring concepts (which are connected via a relation) are similar. Several definitions of the similarity can be considered depending on the scope of the neighborhood considered.

4. Confirm equal or inclusion relationship between concepts

   Concept relationships are confirmed by selecting among equal / including / intersecting / related but no intersecting / no related and no intersecting, by using similarities which we have clarified. If there are inclusion relationships, which concepts should be left after integration are determined.

5. Discover conflicts of relations

   When R_1 is a relation which connects C_1 and C_2, and R_2 is a relation which connects C_2 and C_3, we can see R_1 and R_2 together as one combined relation. The direction of the relation is not pertinent, because if there is a relation in one direction, there is also an implicit relation in the opposite direction. Similarly a combined relation can be defined for more than three relations. In this phase, candidates of equal combined relations between schemas are discovered. Two concepts in one conceptual graph are selected, and equal (or related by inclusion) concepts are searched in the other schema (This can be done, because equality / inclusion relationships are confirmed.). Combined relations between two concepts in both conceptual graphs are candidates for equal relations.

6. Resolve conflicts of relations

   By using decisions made by humans, we confirm that those combined relations are equal / including / intersecting / related but no intersecting / no related and no intersecting, as the case of concepts. If relations are related by inclusion relationship, which relations should be left after integration are determined.

7. Join and simplify conceptual graphs

   "Join" is an operation to derive a new merged graph from two conceptual graphs. Procedures for join are as follows:

   - Merge identical concepts
   - Create the same relations between merged concepts as before

   "Simplify" is an operation for identifying redundant relations as consequences of join. In this phase, we apply the simplify for combined relations which have been determined to be the same.

8. Translate a merged conceptual graph to a schema of the ER model(re-creation)

   We'll discuss details of re-creations in section 5.

4. DATA MODEL TRANSLATION FROM ER MODEL TO CONCEPTUAL GRAPH

4.1. Issues of Existing Methods

Creasy and Ellis discussed a method for using conceptual graphs as a common data model [5]. In their paper, outlines for translating schema in the ER model to conceptual graphs are described. This method has two main issues.

4.1.1. Relationship Type Translation. The first is an issue concerning relationship type translation. One relationship type is simply mapped to one relation. However it is not clear as to which "roles" are mapped. In conceptual graphs relations have directions, but directions of the mapped relations are also unclear.

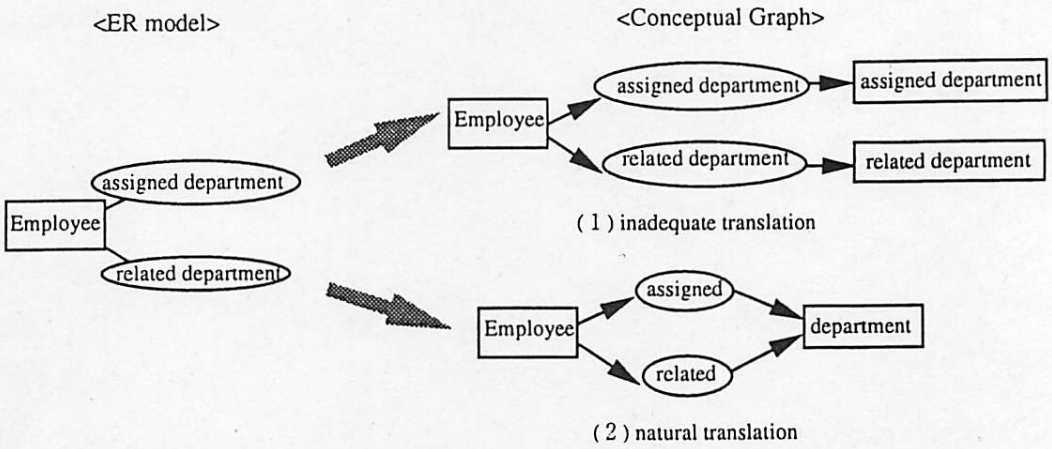


Figure 1. Examples of translations from ER model to conceptual graphs.

4.1.2. Attribute Translation. Attributes in the ER model are mapped to relations and concepts which have the same names as the attributes. We cannot create adequate concepts using this method (figure1) in many cases. For example, an entity type "employee" has attributes "assigned department" and "related department". If we translate those attributes using this method, two different concepts "assigned department" and "related department" (translation 1) are created. However these two concepts should be one concept "department". It is natural that "assigned" and "related" are interpreted as relations between concepts "employee" and "department"(translation 2). If we integrate the translated schema of translation 1 and a schema in which "department" is modeled separated, "assigned department" and "department" create a naming conflict.

The reason why such situations occur is that an attribute of the ER model is a mixture of "a concept to represent the real world" and "a dependency relationship between concepts". We should separate this mixture when translating the ER model to conceptual graphs.

4.2. Concept Creation Rules from Entity Types and Relationship Types

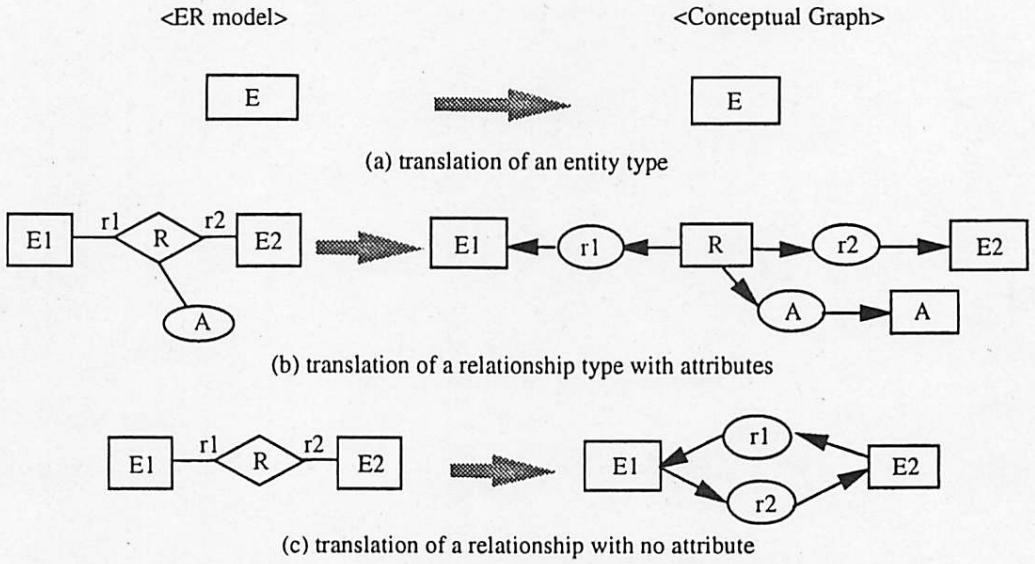
The following shows how to create concepts in conceptual graphs from entity types and relationship types (figure 2).

1. Entity Types

   Create a concept which has the same name as the entity type.

2. Relationship Types

   The basic rule is that roles are mapped to relations, because the probability that a relation created from an attribute and a relation created from a relationship are equal is high. An attribute represents the role of a domain between a participating entity type and the domain. Namely both "a relation created from an attribute" and "a relation created from a role" represent "roles", so there is high probability that they are equal. For example, please consider a case in which industrial parts are modeled. If a model is created using relationships, relationship type "parent_children", role "parent_parts", and "child_parts" are created. If a model is created using attributes, attribute "parent_parts_number", and "child_parts_number" are created. So roles and attributes are to equate. Roles should be mapped to relations.



(a) translation of an entity type

(b) translation of a relationship type with attributes

(c) translation of a relationship with no attribute

Figure 2. Concept creation rules from entity types and relationship types.

(a) CASE 1: Relationship type has attributes.

Since there is no modeling construct in which a relation connects a relation and a concept, a relationship type is mapped to one individual concept having the same name as the relationship type. We create a relation between concepts created from entity types which were connected to the relationship type. Then the direction of the relation is from "a concept whose origin is a relationship type" to "a concept whose origin is an entity type."

(b) CASE 2: Relationship type has no attributes.

Let two concepts, mapped from entity types which ware connected to the relationship type R, be C_a, C_b. Then we create

- a relation R_1 whose direction is $C_{E_2} \rightarrow C_{E_1}$ and with the role name of E_1
- a relation R_2 whose direction is $C_{E_1} \rightarrow C_{E_2}$ and with the role name of E_2.

An interrelation between the relationship and the roles are represented by type-hierarchy ($R_1 \leq R, R_2 \leq R$). This type-hierarchy is a special one which is different from a normal inclusion relation of a relationship type, so this is separated from a schema (i.e. conceptual graphs of origins).

4.3. Rules for Concept Creation from Attributes

An attribute decomposition method is applied for solving the problem in which "concepts" and "relations between concepts" are not separated in the ER model. In general, techniques of natural language processing can be used, however here only concept creation rules when attribute names are decomposed by certain semantic rules are discussed. This paper assume that attribute names obey the following naming rule as the semantic decomposition.

4.3.1. Durell's Naming Rule. For a purpose of standardization of data items such as attributes of entity types, Durell has proposed a naming rule for data items [6]. These naming rule is as follows. A name of a data item is defined as a combination of four

elements, (modifier word of prime word: MP) + (prime word: P) + (modifier word of class word: MC) + (class word: C).

Modifier Word A word modifying a prime word or a class word. This is optional (greater 0).

Prime Word A word for an object to define. This is a necessary element.

Class Word A word indicating an aim of a data item. Examples are code, number, and date. This is a necessary element.

For example, a name of a building which is the starting point of a circuit is "start-point_build-ing_name". ("_" is used as the delimiter of elements.)

4.3.2. Concept Creation Rules from Attributes Obeying the Naming Rule. Here is a description of the concept creation rules from attributes obeying the naming rule. Let an attribute of an entity type E be A. Assume the name of A obeys Durell's naming rule, that is

A = (modifier of prime word) + (prime word) +(modifier of class word) + (class word) $= MP + P + MC + C$.

First basic ideas are described then the rules for creation.

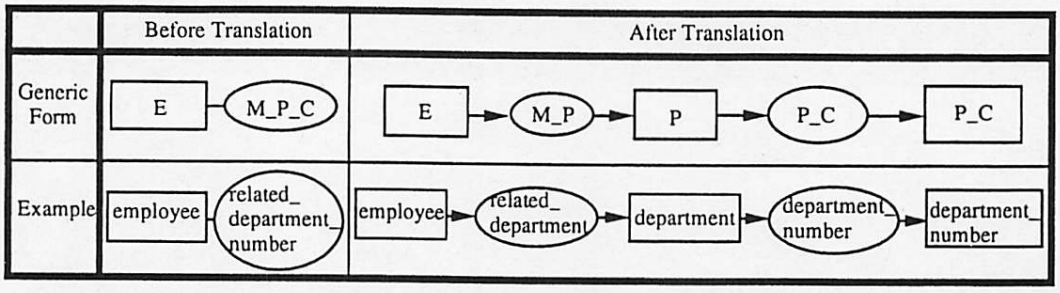
A prime word is a candidate for a concept. However in many cases a prime word is equal to a name of the entity type which has the attribute. Then if a concept is created from the prime word, this concept and the concept created from the entity type name are redundant. Therefore it is not created.

A modifier word of a prime word often means the relationship between the prime word and an entity type that the attribute belongs to, so it is transformed into a relation. There may be no modifier because it is an option. When there is no modifier, if an entity type name and a prime word are not identical, a prime name is mapped to a relation name. If they are identical, an attribute name is mapped to a relation name.

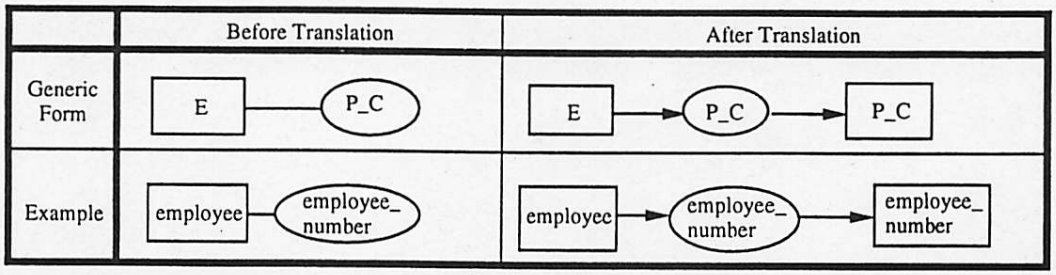
A class word does not represent a single concept, so it is not mapped into a single concept. Similarly, MC, which includes a modifier, is not mapped into a single concept.

The Rules based on these ideas are as follows. MC is omitted because of the above reason.

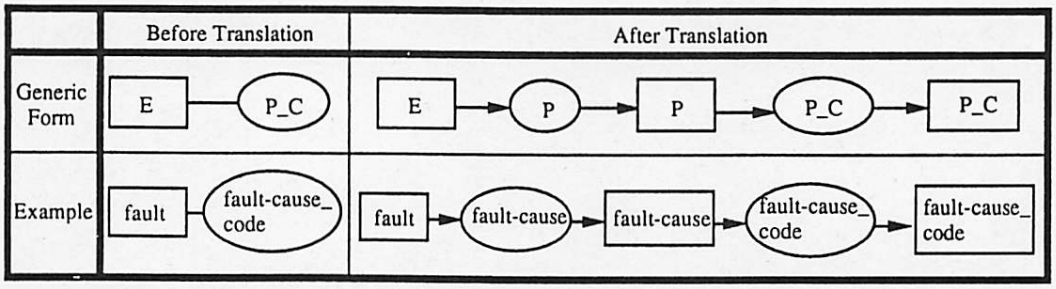
```
/*** Concept Creation Rules from Attributes ***/
/* CASE 1: A modifier exists (figure 3(a)) */
if (M exists)
    if (concepts P,P_C have not been created)
        create P,P_C;
    else
        create a relation M between E and P;
        create a relation P_C between P and P_C;
    end
end
if (M does not exist)
    /* CASE 2: no modifiers and P = E (figure 3(b)) */
    if (P = E)
        if (concept P_C has not been created)
            create a concept P_C;
```



(a) CASE1 : MP exists



(b) CASE2 : no MP exists, and E=P



(c) CASE3 : no MP exists, and E≠P

Figure 3. Concept creation rules from attributes.

```
        end
        create a relation P_C between P and P_C;
    end
    /* CASE 3: no modifiers and P ≠ E(figure 3(c)) */
    else if (P ≠ E)
        if (concept P,P_C have not been created)
            create P,P_C;
        end
        create a relation P between E and P;
        create a relation P_C between P and P_C;
    end
end
```

5. RE-CREATION OF ER STRUCTURES = TRANSLATION FROM CONCEPTUAL GRAPHS TO ER MODEL

The way to re-create ER structures from integrated conceptual graphs is described here. The problem as to which data model construct should be used arises (This also arises when designing from scratch using the ER model.). There is no clear solution for deciding which construct is really best. Therefore we choose a strategy in which original schemes are integrated and kept in an integrated schema.

5.1. Origins of Concepts and Relations

An "origin of a concept (or relation)" is defined as a modeling construct which was used to represent the concept (or relation) in the original ER schema.

Origins of a concept can be one of the following four constructs:

- an entity type (E)
- a relationship type (R)
- a prime word of an attribute (P)
- a prime word of an attribute (+ modifier for a class word) + class word (P_C).

On the other hand, origins of a relation can be one of the following four constructs,

- a role (r)
- a modifier for prime word of an attribute (M)
- a prime word of an attribute + class word (P_C)
- a prime word of an attribute (P).

When two conceptual graphs are joined, there are at most two origins of a concept (relation). A concept (relation) may have only one origin in one original schema (there is no origin in the other schema). We use "ϕ" to represent that there is no origin in an original schema, and "$*$" to represent an arbitrary origin. We indicate a pair of origins by (origin of schema1, origin of schema2). An origin of a concept C is indicated by $org(C)$.

5.2. Re-creation Rules of Concepts and Relations

A basic rule of re-creating concepts is that a concept, which was an entity type, is re-created to an entity type in an integrated schema. After re-creation, an integrated schema must satisfy the constraints of the ER model such as that a relationship type is connected to at least two entity types, or that a relationship type is never directly connected to a relationship type. If an origin of a concept is (relationship, part of attribute), it must be re-created to an entity type, because if it is re-created to a relationship type, a degree of the relationship type changes. This result in a change in the semantics of relationship.

A basic rule for re-creating relations is when both concepts, which connect to a relation, are re-created to entity types, the relation is re-created to a relationship type. So the re-creation of concepts must be done before re-creating of relations. Such orders of re-creation are included in an algorithm in the next section.

5.3. Re-creation Algorithm

The following is an outline showing an algorithm for re-creating ER structures that follow the rules described in the previous section.

A concept is indicated by C and a relation by R. A concept whose origin includes an entity type is indicated by C_E, a re-created entity type from a concept C is indicated by E_C (whose name is C). So if a concept is re-created, whose origin is an entity type, to an entity type, the re-created concept is indicated by E_{C_E}. $C \gg E_C$ is used when the concept C is re-created to an entity type E_C. There is no order in an origin that is (A,B) means both (A,B) and (B,A).

/* step 1: one origin is an entity type → entity type */
for C_E s.t. $org(C_E) = (E*)$
  create an entity type E_{C_E};
end
/* step 2: origins are a relationship type and an attribute → entity type */
for C_R s.t. $org(C_R) = (RP\_C)$ or (RP)
  create an entity type E_{C_R};
  for R_r s.t. $[\exists C_E]$-(R_r)->$[C_R]$, $R_r \leq C_R$
    create a relationship type R_r between E_{C_R} and E_{C_E};
end
/* step 3: origins are relationship types → relationship type */
for C_R s.t. $org(C_R) = (R\phi)$ or (RR)
  /* $[C_1]$-(R)->$[C_2]$ */
  create a relationship type R_{C_R} between E_{C_1} and E_{C_2};
end
/* step 4: a relation between concepts which are re-created to entity types → relationship type */
for R s.t. $[\exists C_1]$-(R)-> $[\exists C_2]$, $C_1 \gg E_{C_1}$, $C_2 \gg E_{C_2}$
  if $(org(R) = (r*))$
    if (a relationship type R_R has not been created)
      create a relationship type R_R between E_{C_1} and E_{C_2};
  else if $(org(R) = (M,*), (P\_C,*)$, or $(P,*)$
    create a relationship type R_R between E_{C_1}, E_{C_2};
end

/* step 5: re-creation of a concept whose origin is a prime word */
for C_P s.t. $org(C_P) = (P\ )$ or $(\quad)$
/* $[C_1]$-(R_1)->$[C_P]$->(R_2)->$[C_2]$ */
if ($C_1 \gg E_{C_1}$, $C_2 \gg E_{C_2}$, and R_1 or R_2 is a key)
create a relationship type R_{C_P} between E_{C_1} and E_{C_2};
if ($C_1 \gg E_{C_1}$, $\sim$ ($C_2 \gg E_{C_2}$ and R_1 or R_2 is a key)) /* $\sim$ means negation */
create an attribute $A_{C_{P\_C}}$ of an entity type E_{C_1};
end
/* step 6: re-creation of concepts whose origins are a prime word + a class word */
for C_{P_C} s.t. $org(C_{P\_C}) = (\ \_C\ )$ or $(\ \_\quad\_\ )$
/* $[\ _P]$-(R)->$[\ _{P\_C}]$ */
if $(\ _P \gg E_C)$
create an attribute $A_{C_{P\_C}}$ of an entity type E_C;
end

6. APPLICATION EXAMPLE

An application example of this method is described. Comparisons of similarities are omitted for simplicity. This example concerns schemas for managing telecommunication circuits and their clients who pay for the circuits (figure 4).

Here schemes to be integrated are termed "local schemes". In LS1 (abbreviation for Local Schema 1 in figure 4(a-1).), there are two entity types "circuits" and "clients". They are connected by a relationship type "payment request". Buildings are managed as an attribute of "circuit". In LS1 an entity type "client" includes both clients of 'sum payment request', which is a payment request of several circuits in a lump sum, and normal clients. In LS2 (figure 4(a-2)), there are three entity types "circuit", "client", and "building". For a sum payment request, client addresses of "circuit" are managed by a relationship type "sum payment request" and an attribute "client address" of entity type "client". For normal payment request, client addresses are managed by an attribute "client address".

1. Translation to conceptual graphs

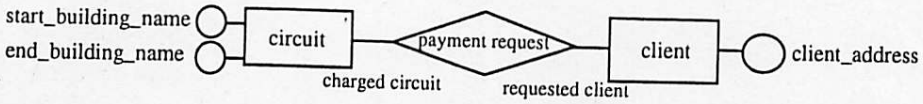
   Results of translating LS1 and LS2 are shown in figure 4(b-1) and (b-2). An attribute "start_building-name" and "client_address" are translated using CASE 1 and CASE 2 of rules in 4.3.2, respectively. We distinguish two concepts for "client" and two concepts for "client_address" (Represented by "client #1" and "client #2".).

2. Computing similarity of concept and decision of equality or inclusion

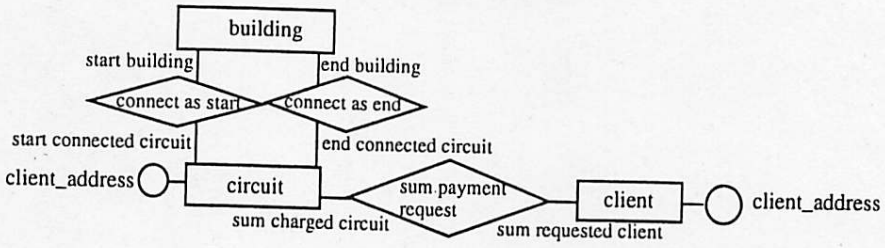
   For example "building"s in two conceptual graphs are determined to be equal. We can determine that LS2.client #1 $\leq$ LS1.client, and LS2.client #2 $\leq$ LS1.client by analyzing the meaning of clients. Therefore "client #1" is renamed to "sum payment client", and "client #2" is renamed to "normal payment client". All concepts, which have an inclusion relation to some concept, are left after integration.

3. Discovering conflicts of relations

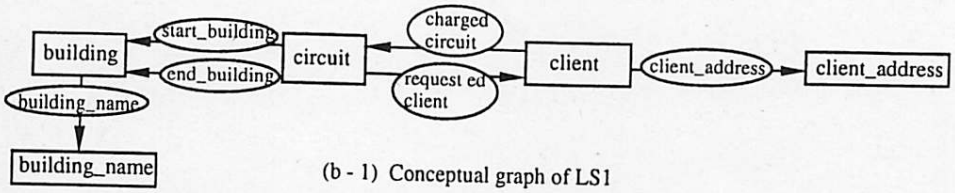
   For example, between concept "circuit" and "client" (the direction is $\rightarrow$) there are candidates of relations to be identified. In LS1 the candidate is the relation "client", in LS2 they are "client" and "sum payment client".



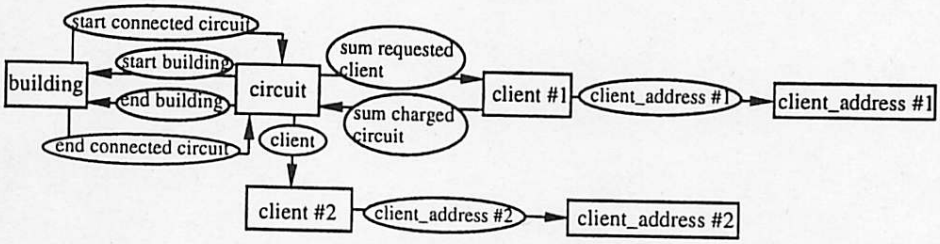
(a - 1) LS (Local Schema) 1



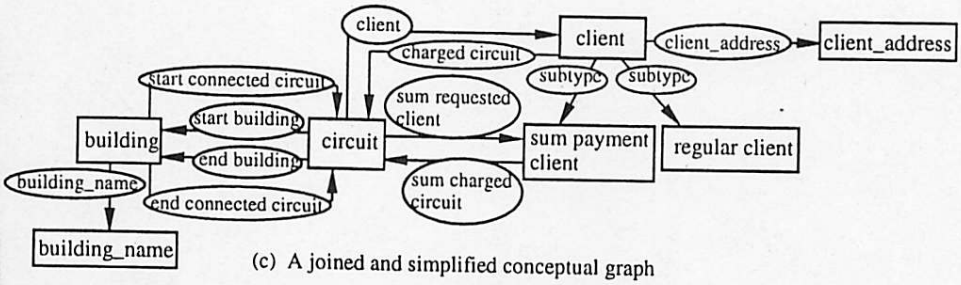
(a - 2) LS2



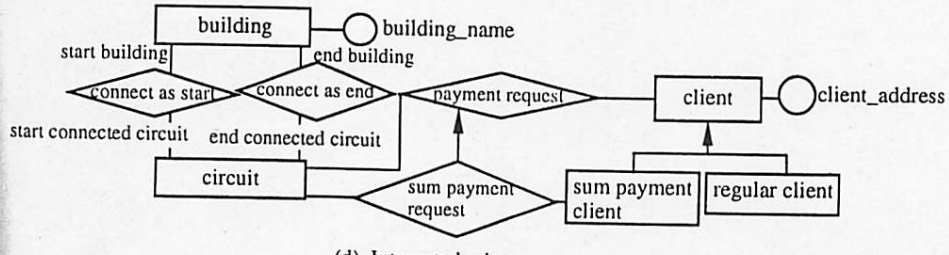
(b - 1) Conceptual graph of LS1



(b - 2) Conceptual graph of LS2



(c) A joined and simplified conceptual graph



(d) Integrated schema

Figure 4. Example of integration.

4. Resolution of conflicts of relations

   For example, a pair of relations "start building" and "start_building" are determined to be equal. If we compare the example of the relations in 3., the results are LS2.client $\leq$ LS1.client, and LS2.sum payment client $\leq$ LS1.client. We decide that LS2.client is not left after integration because it is redundant.

5. Join and simplify the conceptual graphs

   The two conceptual graphs in the previous step are joined and the identified relations are simplified. A result is shown in figure 4(c).

6. Translating to the ER Model

   "Building", "circuit", and "client", which were entity types, are translated to entity types. The relationship "payment request" and "sum payment request" are combined by an arrow of generalization hierarchy. The result is shown in figure 4(d).

7. CONCLUSION

We have shown a method for schema integration using conceptual graphs. Using this we have clarified a methodology including similarity checking and structural conflict resolution, a data model translation algorithm to generate adequate concepts, and a method for re-creating an ER model from integrated conceptual graphs.

REFERENCES

1. Batini, C. and Lenzerini, M., "A Methodology for Data Schema Integration in the Entity Relationship Model," *IEEE Transaction on Software Engineering*, 1984. (6).
2. Batini, C., Lenzerini, M., and Navathe, S.B., "A Comparative Analysis of Methodologies for Database Schema Integration," *ACM Computing Surveys*, 1986. (4).
3. Bouzeghoub, M. and Comyn-Wattiau, I. "View Integration by Semantic Unification and Transformation of Data Structures," in *Proc. International Conference on Entity-Relationship Approach*, 1990. Lausanne, Switzerland: North-Holland.
4. Chen, P. P. "The Entity Relationship Model - Toward a Unified View of Data," *ACM Transaction on Database Systems*. 1976. (1).
5. Creasy, P. and Ellis, G. "A Conceptual Graphs Approach to Conceptual Schema Integration," in *Proc. International Conference on Conceptual Structures*. 1993. Springer Verlag (Lecture Note AI 699).
6. Durell, W. R., *Data Administration (Japanese translation)*. 1987 (original 1985), Nikkei-BP-sha.
7. Hammer, M. and McLeod, D., "Database Description with SDM: A Semantic Data Model," *ACM Transaction on Database Systems*, 1984.
8. Hayne, S. and Ram, S., "Multi-User View Integration System (Muvis): an Expert System for View Integration," in *Proc. IEEE International Conference on Data Engineering*. 1990. Los Angeles.
9. Kitagawa, T. and Kiyoki, Y., "A Mathematical Model of Meaning and Its Application to Multidatabase Systems," in *Proc. IMS'93*, 1993.
10. Larson, J. A., Navathe, S. B., and Elmasri, R., "Theory of Attribute Equivalence in Databases with Application to Schema Integration," *IEEE Transaction on Software Engineering*, 1989. (4).
11. Sheth, A. P. and Larson, J. A., "Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases," *ACM Computing Surveys*, 1990. 22(3).
12. Sowa, J. F., *Conceptual Structures: Information Processing in Mind and Machine*. 1984, Addison-Wesley.
13. Spaccapietra, S. and Parent, C., "View Integration: a Step Forward in Solving Structural Conflicts," *IEEE Transaction on Knowledge and Data Engineering*, 1994.
14. Suzuki, G., Yamamuro, M., Nakawatase, S. "The Method for Discovering Similar Schema Elements in Schema Integration" in *Proc. IEICE Fall Conference*. (in Japanese), 1994. Sendai, Japan.

ER 2001

20th

27 - 30 November 2001

CONCEPTUAL MODELING

Yokohama JAPAN

Industrial Presentations

20th International Conference on Conceptual Modeling

ER 2001

Yokohama National University Education and Culture Hall
Yokohama, Japan, November 27-30, 2001

Industrial Presentations

Applications (11:15-12:45, November 29, 2001)

Software Engineering (09:15-10:45, November 30, 2001)

MeatM PrestoM

A Mediation System Based on Universal Relation Modeling

Takashi HONISHI, Gengo SUZUKI,
Nobuyuki KOBAYASHI, Kazuya KONISHI
NTT Cyber Space Laboratories
{honishi, gsuzuki, kobayasi, koni}@dq.isl.ntt.co.jp

MeatM PrestoM

Information Sources Integration System

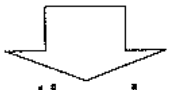
System Architecture of Integrating Information Sources

- Federated Database Architecture
  - Based on an integrated schema
  - But, Making the schema is extremely difficult
- Our Mediation Architecture
  - Dynamically make an integrated schema from setting information (mediation schema)
  - Mediation schema independent of all information sources
  - And, Making the schema is easy

Background

Necessity of an Information Source Integration System

- Huge numbers of information sources on the Internet and intranets
- Heterogeneous information sources
  - Structural conflict
  - Naming conflict
  - Etc...



- Information integration becomes key technology

Technological Problems

Differences between Information Sources

- Schema Structures
  - Schema structure defined in each information source
  - (ex) Relation(RDB) or Tree structure(XML)
- Schema Definition names
  - Item name in each individual system
  - (ex) "Price" or "Cost"
- Data Range
  - Value for items with identical meaning
  - (ex) "¥1000" or "$8.05"

Approach (1)

Resolving the Difference of Schema Structures

- Universal relation :The relation includes all data of all DB
  - (ex)
    Mediation Schema: Reference "Product"

Reference	Data Item	Table	Database
Refer	Purchase	Customers	Products DB
Referred	ID	Products	Products DB

  - The system joins tables "Customers" and "Products", and the user can retrieve all data of both tables

Approach (3)

Resolving the Difference of Data Range

- Domain to which data belongs
- Transformation functions from each data form in the domain to common form of the domain
  - Join or separate data
  - Arithmetic operation for data
    (ex) from "m" to "km"
  - Transform data system
    (ex) from "Japanese" to "English"

Approach (2)

Resolving the Difference of Schema Definition Names

- Dictionary of synonyms, and Relations between data items
  - (ex)
    Mediation Schema: Relation "Salary"

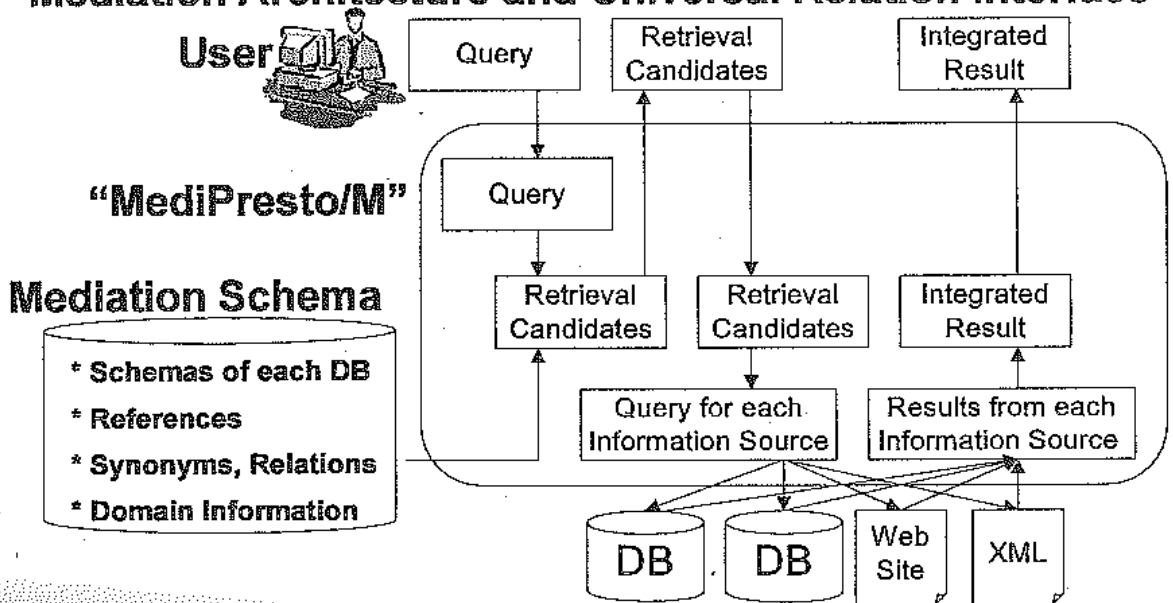
Data Item	Table	Database
Salary	Tokyo Office	Employees DB
Pay	Yokohama Office	Employees DB

  - The system processes data items "Salary" and "Pay" as the same item, and the user can retrieve both data items at once

Our Development System (1)

"MediPresto/M"

Mediation Architecture and Universal Relation Interface



Our Development System (2)

"MediPresto/M"

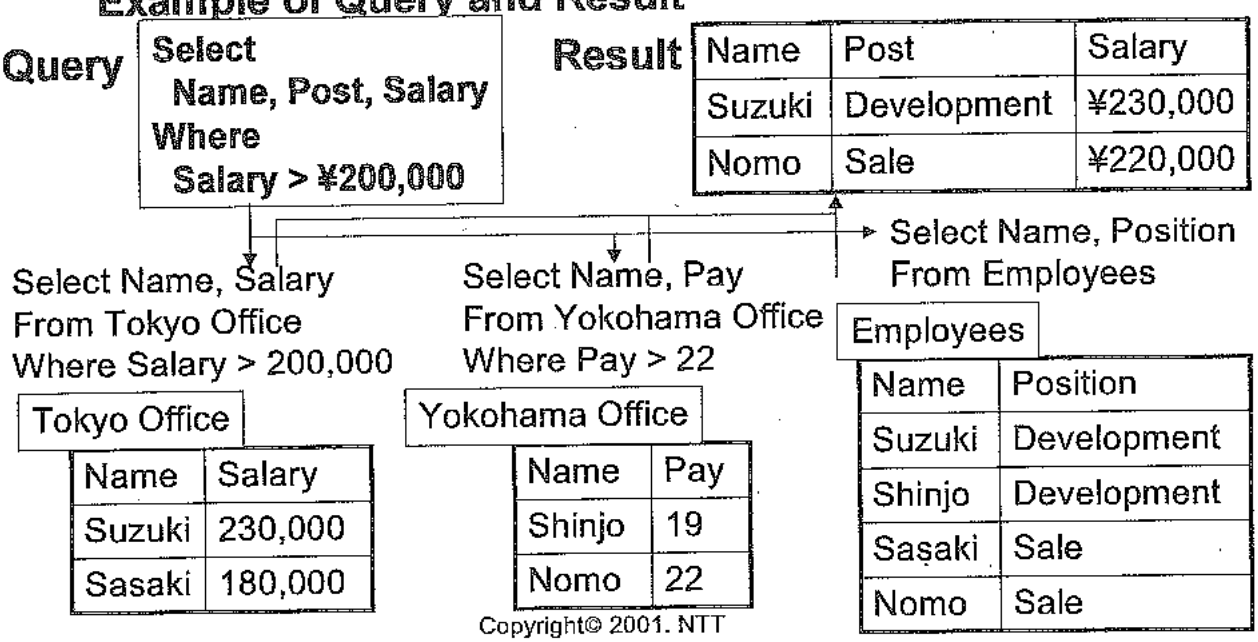
Example of Query and Result

Query

```
Select
  Name, Post, Salary
Where
  Salary > ¥200,000
```

Result

Name	Post	Salary
Suzuki	Development	¥230,000
Nomo	Sale	¥220,000



Name	Position
Suzuki	Development
Shinjo	Development
Sasaki	Sale
Nomo	Sale

Name	Salary
Suzuki	230,000
Sasaki	180,000

Name	Pay
Shinjo	19
Nomo	22

Setting for Mediation Schema

MediPresto/M Modules for Setting several Items

- Collect Schemas of each DB
  - Automatic Collection Function
- Specify Relations between Tables and Data Items
- Specify Join Conditions between Tables
  - GUI Support for specifying relations and references
- Transform Data into Common Form
  - Offer standard transformation function

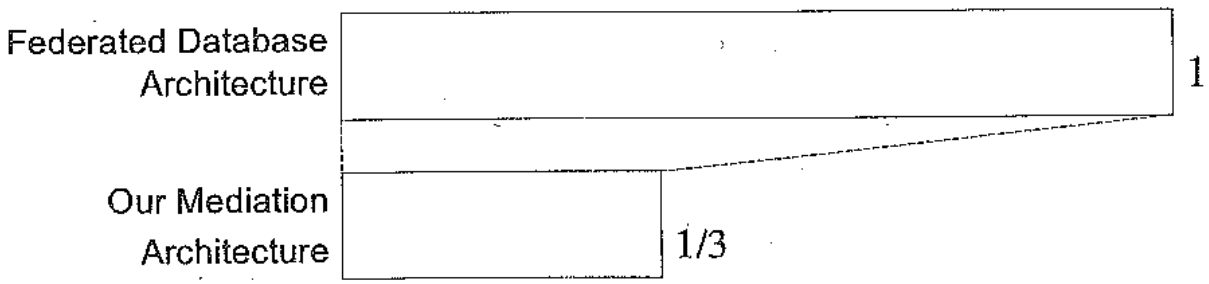
Comparison of Making Schema

Setting for Making Integrated/Mediation Schema

Federated Database Architecture	Mediation Architecture
Collect Schemas of each DB	Collect Schemas of each DB
Transfer Schemas to Common Data Model	
Specify Relations between Tables and Data Items	Specify Relations between Tables and Data Items
Design an Integrated Schema	Specify Join Conditions between Tables
Transform Data to Common Form	Transform Data to Common Form

Evaluation

Compare: The Man Hours Needed to Create Schema



- The man hours needed to create mediation schema
  - About 1/3 when creating the integrated schema

Demonstration

Demonstration of Our System "MediPresto/M"

MediPresto/M

Suggested Applications

- Educational material Retrieval System
  - Retrieve educational materials on Internet for teachers
- Portal Service for Multimedia Archive
  - Integrated image archives, music archives, and movie archives
- Integrated E-Commerce Service
  - Integrated EC sites on Web, i-mode, and L-mode

Actual Applications

- Integrated Multi-Archive Service (IMAS)
  - Integrated information retrieval service in a university library (10 information sources)
- Hospital Information Service
  - Work list service for a doctor (Reservation DB, Inspection image DB, Diagnosis DB)
- R&D Retrieval Service
  - Integrated retrieval from Web sites on industrial technology (1 DB, 1 Web Site, 4 SGML)

Future Work

- Update data via the universal relation interface
- Integration of multimedia data
- Mediation architecture based on XML model (for various information sources)

Chapter 10

Integration of Keyword-Based Source Search and Structure-Based Information Retrieval

Gengo Suzuki, Yuichi Iizuka, Shiro Kasuga
NTT Cyber Space Laboratories
1-1 Hikari-no-oka Yokosuka-Shi Kanagawa, 239-0847 Japan
Email: {gsuzuki, iizuka, kasuga}@dq.isl.ntt.co.jp

Abstract: We propose a new method of accessing heterogeneous information sources such as Web pages and relational databases. In our method, users input only keywords. These keywords are then mapped to appropriate information resources such as values and data items, and then queries to information sources are generated. This approach is a mixture of keyword based web search engines and mediators. The proposed method is described along with a prototypical implementation.

Key words: Accessing heterogeneous information, keyword based search, mediators, information resource dictionary, conformity rate

1. INTRODUCTION

As a result of the explosive spread of the Internet, we now have access to huge numbers of information sources. These sources include web home pages, relational databases, and image databases. Information retrieval has become the key technology of the Internet age.

Most information search services use keyword- based web search engines such as Yahoo and AltaVista. These engines accept free keywords as input and the outputs are the URLs of Web documents. Another approach, called "mediator" systems, has been developed in the database research area[1][4]. A mediator system can integrate several heterogeneous information sources. Mediators have expressive query language interfaces, such as SQL, but are not so easy to use.

H. Bestougeff et al. (eds.), Heterogeneous Information Exchange and Organizational Hubs, 149–158.

This paper proposes a new method of accessing heterogeneous information sources such as Web pages and relational databases. In our method, the keywords input by users are mapped to appropriate information resources such as values and data items, and then queries to information sources are generated. This method lies between keyword based web search engines and mediators.

Section 2 describes our motivation, and the proposed method is detailed in section 3. Section 4 introduces a prototype implementation. Conclusions are given in section 5.

2. MOTIVATION

Table 1 compares the two most common tools for information retrieval. Dominating Internet information search is the URL search engine as used by Yahoo or Infoseek. If users input keywords, the tool looks through all indexed web pages and shows lists of relevant URL addresses.

Mediator systems, another search tool, were proposed by database researchers. Mediators are directed towards multi-database research activities. Users can access various information sources such as web pages or databases by posing query language requests. Retrieval results can return in either table form or object form. Retrieval conditions can, of course, be added. URL search engines are better in terms of input ease, while mediators are better at handling different information sources or structural retrieval.

Our proposed method offers both input ease and structural retrieval. It combines the approaches of full-text search and database retrieval.

	URL search engines	Mediators
Input ease	Good (Keyword)	Not good (Query Language)
Information source coverage	Not good (Only web)	Good (Web, Database,etc)
Structural retrieval support	Not good (Only URLs)	Good (Tables and Objects)
Examples	Yahoo, Infoseek	TSIMMIS (Stanford Univ.) Information Manifold (AT&T)

Table 1. Comparison of Existing Technologies

3. OUR METHOD: KEYWORD INPUT

We propose a retrieval method that offers the merits of both web search engines and mediators. The user interface of this method is keyword input.

The method maps keywords to "data" or "metadata" of information sources, as appropriate. It produces candidate queries that suit the known information sources. Candidate queries are ranked by conformity rates. In the case of web search engines, keywords are mapped only to "data"; in the case of mediators, users must determine whether the keywords map to "data" or "metadata". In our method, users need not be concerned as to whether keywords are "data" or "metadata".

For example, if a user enters the keywords "price" and "BMW", the user's requirement may be "to get the price of BMW cars".

The proposed method can determine that "price" is metadata and "BMW" is data.

If the information source is a relational database, the SQL query "select price from car where car_type like '%BMW%'" would be produced.

The features of our method are as follows:

- Automatic mapping from input keyword to database resources (values, columns, tables, etc.)
- Query candidates are ranked by conformity rate.
- Suitable query generation for each information source (SQL etc.)

When using URL search engines, keywords are mapped to only values. When using mediators, keywords are mapped to only columns and there is no conformity concept.

3.1 System Architecture

Figure 1. shows the system architecture of our method. The key to our method is the *Information Resource Dictionary*.

The *resource discovery module* uses the Keyword Index found in the Information Resource Dictionary.

The *format conversion module* uses the Schema Dictionary and the Domain Dictionary to handle expression differences.

The *query translation module* uses the Schema Dictionary and the Term Dictionary.

Differences in management systems such as DBMS, Web pages, or XML files are hidden by wrapper modules.

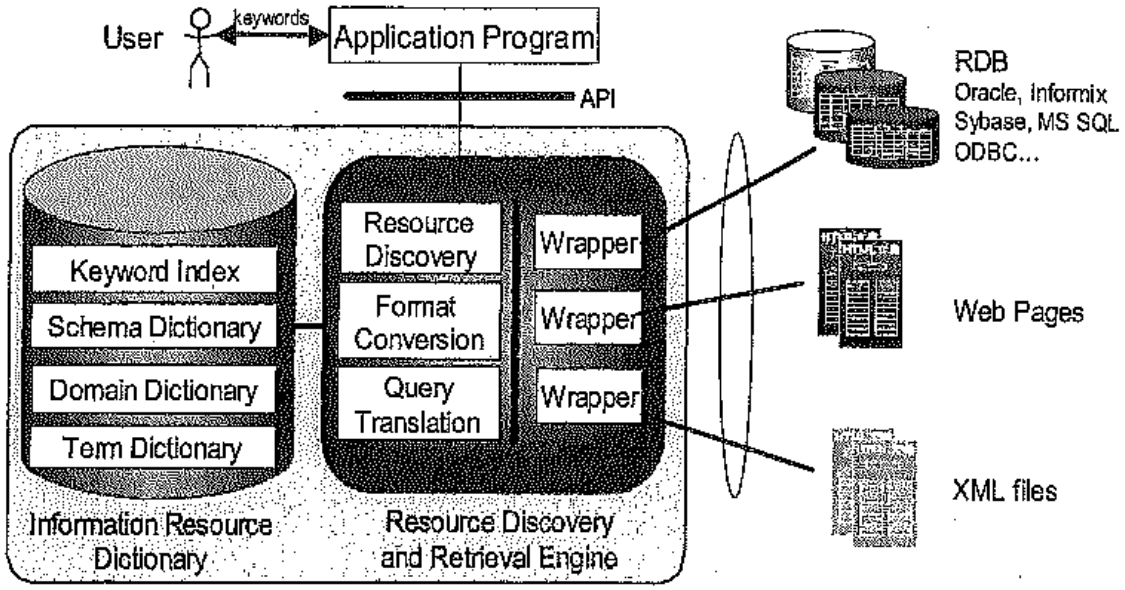


Figure1. System Architecture

3.2 Query Interpretation

The flow of the proposed method is described using the example shown in Figure 2.

The user inputs one or more keywords. In this example, the keywords are *name, Tiyota* and *price.*

First, this method decides whether each keyword is a value, a column name, or a table name. We call this process "resource discovery".

In this example, the keyword "name" matches the column "name", "Tiyota" matches a value in the column "Tiyota", while "price" matches the column "price".

As a result of resource discovery, many resource combinations are produced. We call each combination a "candidate".

The user is presented with candidate lists, he selects the relevant candidates, and executes them. A relevant query for each information source is generated, and the query is sent to the information source. The results are returned to the user.

In this example, the result is the record, Vits, Tiyota, 10000.

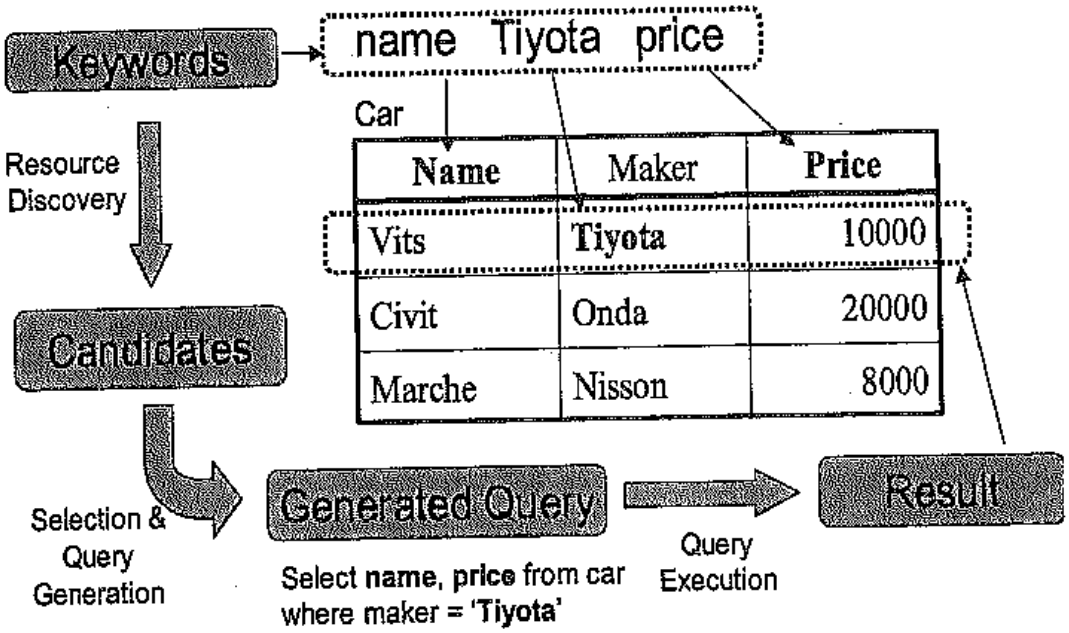


Figure 2. Example of Our Method

3.3 Index Dictionary

To determine keyword location, the index dictionary is used. The structure of the dictionary basically follows the usual inverted index file, but keyword location is described using database resources, not URLs.

Table 2 shows a simplified keyword index dictionary. The key to this table is "keyword". "Type", "Table", and "Column" show keyword location. For example "Tiyota" is one value in the column Maker, which is in the car table of the used-car database. This dictionary can be generated automatically.

Keyword	Type	Database	Table	Column
Tiyota	Value	used-car	car	Maker
Name	data-item	used-car	car	Name
Price	data-item	used-car	car	Price
...	...	...	...	...

Table2. Index Dictionary

3.4 Conformity Rate

Candidates are ranked by a conformity rate. The expressions shown below are used to calculate the conformity rate.

The *hit rate* is the ratio of the number of keywords in one candidate to the total number of keywords. "If a candidate includes many keywords, it's good". The number of keyword hits is weighted by matched resources.

For example, a value match is more important than a meta-data explanation match. Accordingly, the hit rate is maximized if all keywords match the values.

The *distribution rate* of a candidate is inversely proportional to the number of tables, databases and systems. If a candidate has many tables or databases, its distribution rate is low. This means that a "Single site candidate is good"

The *conformity rate* is calculated for the hit rate considering the influence of the distribution rate. In the following expression, the degree of influence of the distribution rate is 10%.

Hit Rate:

$$h = \frac{1}{n} \sum_{hit-keywords} r \tag{1}$$

n: number of input keywords
r: resource parameter per hit keyword (0 to 1)
if value match, then r = 1
if metadata explanation match, then r = 0.15 etc.

– **Distribution Rate**:

$$d = \frac{6}{T + 2D + 3S} \tag{2}$$

T: number of tables in one candidate
D: number of databases in one candidate
S: number of systems (database group) in one candidate.

– **Conformity Rate**:

$$c = h - \frac{1-d}{10} \tag{3}$$

h: hit rate
d: distribution rate.

3.5 Heterogeneity Resolution

In this system, several heterogeneities among information sources are resolved [2, 3, 5].

– Structural heterogeneity:
If keywords are mapped to data items in several tables or information sources, relationships between tables in schema dictionary are located and added to query candidates.

– Data representation heterogeneity:
We use the concept of "domain", which means data representation of data item such as unit of price [2].
The domain dictionary manages domains of information sources and domains for users. For example, Japanese users want to use "yen" domain for price, while information source price domains are "dollars". The format conversion module can offset such domain differences.

– Naming heterogeneity:
Naming heterogeneities of data items are resolved by a synonym term dictionary.

4. PROTOTYPE IMPLEMENTATION

We developed a prototype system based on our method.
This prototype can integrate, XML documents, HTML documents and relational databases such as Oracle, and Microsoft SQL Server.
It runs on Windows NT4.0 and IIS4.0. Microsoft SQL Server is used for dictionary management.

Figure 3 shows an input image of the prototype. The top area is for keyword input. Users input keywords in this area, and push the search button; candidates are shown in the bottom area. The conformity rate is shown on the left. Conditions such as "price is less than $40000" can be added using the text area. Candidate details can be shown by selecting a link. The query is executed when the execution button is pushed

Figure 4 shows details of a candidate. This shows that two information sources are related.
Figure 5 is a result table. In this example, BMWs whose prices are less than $40000 are shown.

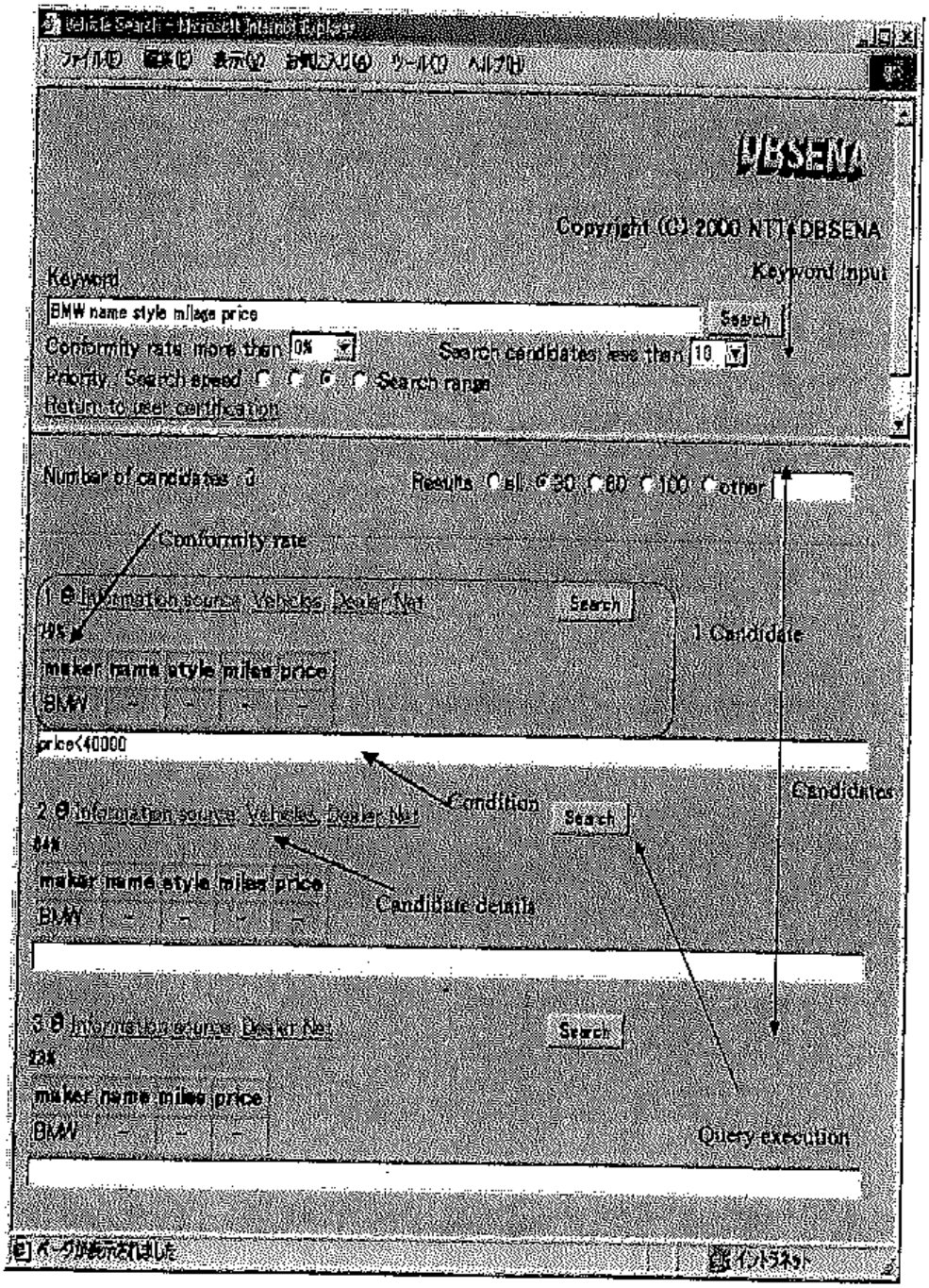


Figure 3. Prototype Image: Input & Candidates

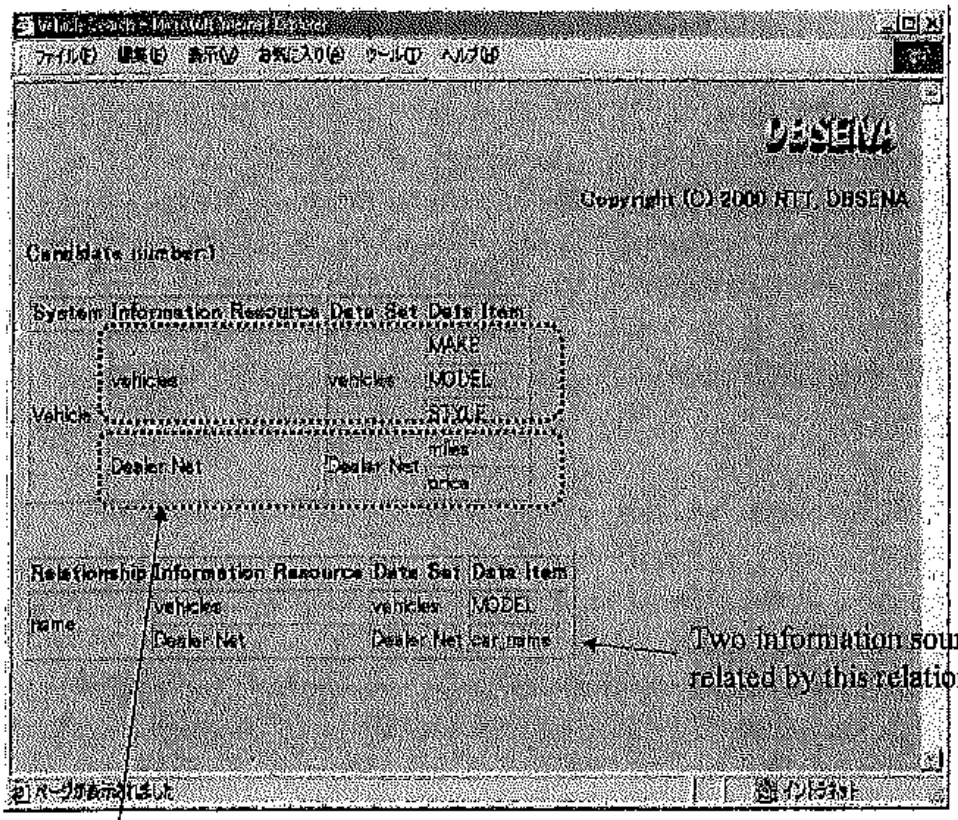


Figure 4. Prototype Image: Candidate Details

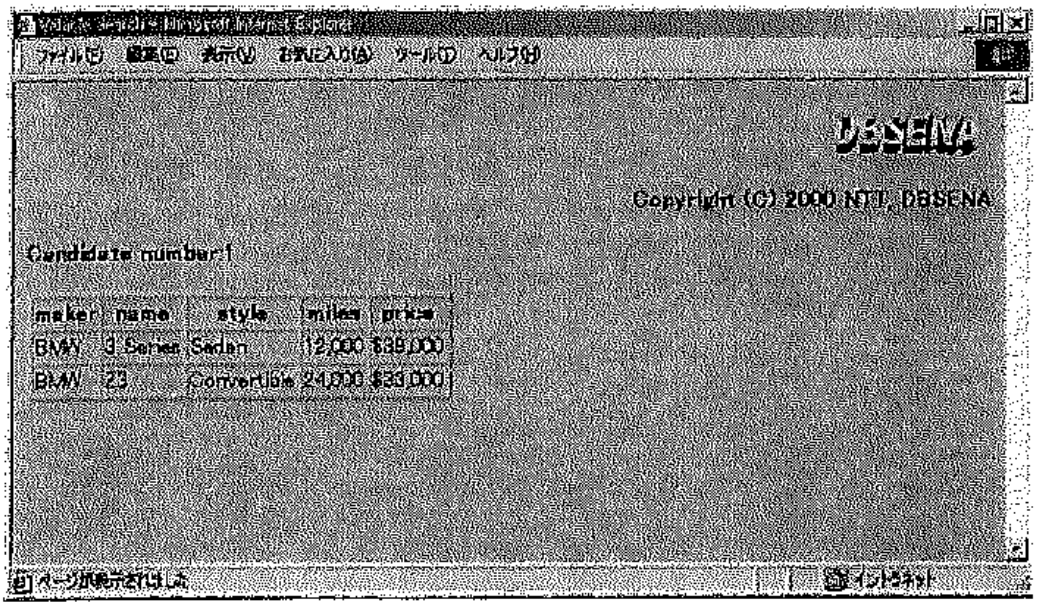


Figure 5. Prototype Image: Retrieval Results

5. CONCLUSION

We proposed an information retrieval method based on information resource management. Our method enables flexible information retrieval against heterogeneous information sources. We developed ways of managing metadata and data, computing conformity rates, and translating keywords to queries. We have implemented a prototype of this method. This prototype can search and integrate various sources such as XML documents, HTML documents, and relational databases.

REFERENCES

1. Domenig R., Dittrich K.R., An Overview and Classification of Mediated Query Systems, SIGMOD Record vol 28, no 3, 1999.
2. Iizuka Y., Tsunakawa M., Seo S., Ikeda T., An Approach to Integration of Web Information Source Search and Web Information Retrieval, ACM Symposium on Applied Computing, 2000.
3. Kim W., Seo J., Classifying Schematic and Data Heterogeneity in Multidatabase Systems, Computer, Vol.24, No.12, pp 12-18, 1993.
4. Levy A.Y., Rajaraman A., Ordille J.J., Querying Heterogeneous Information Sources Using Source Descriptions, International Conference on Very Large Data Bases(VLDB'96), 1996.
5. Suzuki G., Yamamuro M., Schema Integration Methodology Including Structural Conflict Resolution and Checking Conceptual Similarity, in To-yat Cheung et al. (Eds.), Database Reengineering and Interoperability, Plenum Press, New York, pp 247-260, 1996.

情報処理学会論文誌 Vol.53 No.2 857–867 (Feb. 2012)

時間制約を持つ寄り道経路探索システムの実現と評価

鈴木 源吾[1,a)] 榎本 俊文[1] 小林 伸幸[1] 山室 雅司[1] 鬼塚 真[1]

受付日 2011年5月30日, 採録日 2011年11月7日

概要：近年，カーナビ，インターネット等で地図検索，特にルート探索等のサービスが多数提供されている．本論文は，これらのルート探索において，経由地に時間制約があるような最短経路探索問題である「タイムセール寄り道探索」の概念を示すとともに，その解を求める手法を提案する．従来の寄り道探索手法をそのまま利用できる「基本導出法」を示し，その課題を指摘し，さらに，グラフの探索を行いつつ制約条件をチェックし解を導出する「動的導出法」を提案した．動的導出法は，グラフの探索範囲と候補となる解の個数を抑制し，性能を改善させることを特徴に持つ．グラフデータベース上に構築した実験システムを用いて提案方式の比較評価を行い，動的導出法が基本導出法に比べて，特にサービス密度（サービスを実施しているノードの割合）が低い場合に性能的に優れており，探索対象が大規模となる場合において適用性が高いことを示した．

キーワード：グラフデータベース，経路探索

Time Constrained Trip Planning Search System

Gengo Suzuki[1,a)] Toshifumi Enomoto[1] Nobuyuki Kobayashi[1]
Masashi Yamamuro[1] Makoto Onizuka[1]

Received: May 30, 2011, Accepted: November 7, 2011

Abstract: We propose and formalize "time-sale trip planning search", which is a variation of shortest path problem. This is a trip planning which includes time-constrained services. We clarify "basic method" of this search, which is a simple extension of existing trip planning method, and "dynamic method" which can restrict search range and number of candidate answers using time constraints. We implemented these methods on a graph database system, and showed that dynamic method has advantages in performance under low service density (ratio of nodes in service), so that can be applicable for very large graph databases.

Keywords: graph database, route search

1. はじめに

近年，カーナビゲーション，スマートフォン，地図サービス等の普及により，旅行計画作成に関するニーズが増大しており，様々なサービスが実用化されている．最も単純な旅行計画作成の例は，最短経路探索であり，鉄道網や道路網から所望のルートを探索するサービスに活用されている．これは，ノード間の移動コスト（例：所要時間）が与えられたグラフにおいて，合計コストが最小となる最短のルートを求める問題であり，ダイクストラ法，A*法等のグラフ探索技術が確立している [1], [2], [3].

一方，ユーザの位置情報に応じて，お勧めの場所を推薦するレコメンドサービスがある．ユーザの位置（GPS 等により取得）の近くにあり，ユーザの希望，旨向にあった店を推薦したり，目的地までのルートを探索したとき，そのルート沿いのガソリンスタンドやファーストフード店を推薦したり等，交通やグルメ等のインターネット上の検索サービスの分野 [4], [5] や，カーナビ等のナビゲーションシステムで用いられている．

また，上記サービスに関係するが，少し目的が違う，次のような要求もある．たとえば，ユーザがどこかに出張す

[1] NTT サイバースペース研究所
NTT Cyber Space Laboratories, Yokosuka, Kanagawa 239–0847, Japan
[a)] suzuki.gengo@lab.ntt.co.jp

るときに，必ずしも最短経路でなく，多少遠回りしてもよいから，そのルート上で自分の好みの昼食をとってから目的地に到達したい，という要求である．この要求を満たす探索を「寄り道探索」と呼ぶことにする．このとき経由地をPOI（Point of interest）と呼ぶ．ユーザからは，直接具体的なPOIではなく，「寿司が食べたい」とか「銀行に行きたい」というような行きたい場所のカテゴリで指定されることを想定する．具体的なPOIが事前に決まっているのであれば，出発点–POI・POI–到達点の2つのルートを最短経路探索法で求めればよいので，技術的には既存の技術で解決できる．しかし，寄り道探索では，あらかじめ分からないPOIを探索しつつ，できるだけ最短経路に近いルートを求めることが必要になる．

寄り道探索において，寄り道先に時間制約がある場合を考える．たとえば，開店時間には12時～22時のように制限がある．その開店時間に到着できるようなルートだけを結果としてほしい．たとえば，夜遅めに出発するときには，閉店に間に合うように出発点に近いPOIに寄り道する必要がある．また，開店時間だけでなく，「12時～13時，ランチ2割引」や「17時以降，閉店前3割引」のような時間限定のタイムセールのときに店に行きたいという要求も同じ課題であるといえる．このような，時間制約を持つ寄り道探索を「タイムセール寄り道探索」と呼ぶ．上記では，開店時間等のサービスの時間制約を例としてあげたが，「その店で2時間ゆっくりしたい」「銀行でお金をおろすのでぎりぎりに間に合えばよい」というような，ユーザにとっての時間制約の要求もあり，これらも満たすような探索を行いたい．

まず，時間制約を満たす解を導出する方法が課題となる．通常の寄り道探索では総所要時間は，出発点からPOIまでの所要時間と，POIから到達点までの所要時間を単純に足せばよいが，タイムセール寄り道探索の場合，待ち時間も許容することがサービス的に有効である．待ち時間も含めたスケジュールを決定し，その総所要時間で最終的にランキングする必要がある．また，寄り道探索はPOIを探すために，通常の最短経路探索以上に多くのノードを探索する必要があるため，できるだけノードの展開を減らし，探索速度を向上させることも課題である．

本論文では，大沢ら[6]で提案されている寄り道探索のための逐次拡大法をそのまま利用し，それにタイムセール寄り道探索解を導出する方法を組み合わせた基本導出法と，さらに時間制約を積極的にグラフ探索に利用する動的導出法を提案している．動的導出法は，探索範囲と解候補の個数を制限し性能を改善できる．

本論文は以下のように構成されている．まず，2章で過去の研究との関連を示し，3章でタイムセール寄り道探索を定義し，4章で既存の寄り道探索の手法と基本導出法について述べる．5章で動的導出法を示す．6章で提案手法をグラフデータベースを用いて実装した実験システムとその評価を述べ，7章で結論と今後の課題を述べる．

2. 関連研究

本論文で扱っているのは，グラフ上でPOIを探索し，条件付きの最短経路を求める問題であり，グラフデータベース・空間データベースの分野との関連が深い．空間データベース検索の分野では，k-最近傍探索等の分野で類似のPOIを探索する方式が提案されている[7], [8]．その代表例として，Sharifzadehら[7]によるOptimal Sequenced Route探索があげられる．これは，複数の異なるPOIを指定された順序で探索するという一種の最近傍探索でありユークリッド距離を用いている．本論文のようなグラフ上の最短経路探索ではない．また，Optimal Sequenced Route探索は準最適解であることに対し，本論文で提案する手法では最適解を求めることが可能である．**表1**にその違いをまとめた．一方，ユークリッド距離ではなくグラフ上の距離で，最短経路探索等を効率良く求める技術の研究が行われており[9]，特にネットワークボロノイ図等を前処理で作成し効率化をはかる研究がさかんである[10]．しかし，本論文では頻繁に変更される時間制約があるような応用を想定する．たとえば，我々が想定している応用として，レストランやスーパマーケットのタイムセールの探索がある．これは，事前にスケジュールが決まっておらず，当日の在庫状況，顧客状況等によって実施が決定され，それをPOIにいる従業員がtwitterでつぶやくと探索の対象として登録されるというように，実時間性がある．その場合，前処理が必要であるとそのやり直しが頻繁に発生してしまうので適しておらず，前処理を必要としないことが望ましい．たとえば，Sharifzadehら[7]や蒲原ら[10]は，ネットワークボロノイ図を事前に作成しておく必要があり，時間制約を利用する問題に適さない．前処理を必要としない手法として，大沢ら[6]による逐次拡大法がある．これは，POIが1つである制約はあるが，前処理なしで寄り道経路探索を実現している．本論文ではその手法を拡張している．

時間の制約という意味では，時間概念を組み込んだ時制データベース[11], [12]が研究されているが，関係モデルやXML等の一般的なデータモデルに時間概念を組み合わせ

表1 ルート探索手法の比較
Table 1 Comparison of route search method.

	Sequenced Route 探索	タイムセール寄り道探索
利用する距離	ユークリッド距離	グラフ上の距離
出発点指定	あり	あり
到着点指定	なし	あり
寄り道先の数	複数	1
順序性	あり	なし
時間指定	なし	あり
解の最適性	準最適解	最適解

る研究が主流になっている．本論文は，グラフデータベースに時間概念を取り込んでいることに特徴があるが，サービスの概念に特化しているという意味で，一般的な時制グラフデータベースとまではいえない．しかし，今後，一般的な時制グラフデータベースの研究を進めるうえでの最初の1歩であり，重要な応用例になるものと考える．

本論文では，ユーザやサービスによって指定された時間制約を満たす時間区間を求めているが，これは区間スケジューリング問題 [13] に関連がある．区間スケジューリング問題では，要求する区間に重なりがなく，できるだけ多く（長く）要求を実行できる区間を選ぶことが目標になるが，本論文ではPOIを1つに制約し，答えとなる時間区間の個数が3つに限られていることから，問題に特化した初等的な解法で時間区間を求めている．今後，POIを複数に拡張するときに，より一般的な区間スケジュール問題の成果を利用できる可能性がある．

ユーザの嗜好を反映したルートを提示するような研究としては，松田ら [14] があるが，「歩道のある道を歩きたい」等の要求に応じて，コストに重みを設定しルートを計算する方法であり，本論文で問題としているカテゴリに合致するような寄り道を行う探索には利用することができない．

グラフ情報を蓄積・検索する手段として，グラフのデータモデルを直接扱うことができるデータベース管理システムが一般的になりつつあり，グラフデータベースと呼ばれている [15]．実用的に利用できる製品も近年でてきている [16]．本論文でも，システムの実現と評価に利用している．

3. タイムセール寄り道探索

3.1 タイムセール寄り道探索の定義

以下にタイムセール寄り道探索を定義する．寄り道先は1カ所に限定している．$G=(V,E)$ を重みつきグラフとする．V が頂点（ノード）集合，E が辺（エッジ）集合である．エッジへの重みは移動にかかる時間を表すこととする．

定義1（カテゴリ） ノードの種別を表すラベルをカテゴリと呼ぶ．カテゴリが付与されたノードをPOIと呼ぶ．カテゴリは，寄り道先として指定するために使う「コンビニ」「銀行」「中華料理屋」等の指定を意味する．

定義2（時間区間） 時間区間を [開始時刻, 終了時刻] と表現する．以降の説明では，分単位で時刻と時間を表記し，時刻を表すときに年月日を省略する．

定義3（サービス条件） POIにおいてサービスを識別するラベルをサービス内容と呼ぶ（例：営業時間，ランチサービス，商品2割引）．サービス内容とそのサービスが実施される時間区間の組をサービス条件と呼ぶ．一般的にはサービス実施時間は1つの区間とは限らないが（例：昼と夜に営業），以降の解の導出法では1つのサービス実施は1つの区間であることを前提としている部分がある．一般化は今後の課題である，

定義4（ユーザ条件） 以下の5つのユーザ側の時間制約をユーザ条件と呼ぶ．

- 出発点条件：出発点を出発し探索を開始する時刻を含む時間区間
- POI開始条件：POIでサービスを開始する時刻を含む時間区間
- POI終了条件：POIでサービスを終了する時刻を含む時間区間
- 滞在長条件：POIでサービスを受ける時間の長さの最大値・最小値
- 到着点条件：到着点に到着し探索を終了する時刻を含む時間区間

定義5（タイムセール寄り道探索） カテゴリ・サービス条件が与えられている重みつきグラフ $G=(V,E)$ に対して，タイムセール寄り道探索とは，指定した出発点・到着点・カテゴリ・サービス内容・ユーザ条件・解の個数kに対して，以下の条件を満たすグラフ上のルートと結果となる時間区間であるS区間（出発点からPOIまで移動する時間区間），P区間（POIに滞在する時間区間），E区間（POIから到着点へ移動する時間区間）の組について，総所要時間（E区間の最大値 − S区間の最小値）が最小となる上位k組を求めることをいう．

- ルートは，指定した出発点を出発し，指定したカテゴリを付与されたPOIを経由し，指定した到着点に到着する．
- S区間の最小値は出発点条件に含まれる．
- S区間の長さは出発点からPOIまでの移動時間に一致する．
- P区間の最小値が，POI開始条件に含まれる．
- P区間の最大値が，POI終了条件に含まれる．
- P区間はPOIにおける指定したサービス内容のサービス条件に含まれる．
- P区間の長さが，滞在長条件の最小値と最大値の間にある．
- E区間の最大値が到着点条件に含まれる．
- E区間の長さはPOIから到着点までの移動時間に一致する．

図1 に，ユーザ条件・サービス条件とそれを満足する解の例を示す．タイムセール寄り道は以下にあげる特徴を持ち，単に時間制約を満たすルートを求めるだけでなく，「スケジュールを立案する」といった性格を持っている．

- 区間の位置は一意的に決まるとは限らない．P区間は，この条件であれば，図1に示したように19時半〜21時半でもよいし，19時〜21時としてもよい．移動時間が長ければ，それにより制約を受けるが，決定には任意性がある．その決定のためには，スケジュールに関する何らかの戦略（できるだけ早く物事を済ます，

等）が必要となる．

- POI 滞在区間の前後の待ちを許容している．S 区間と E 区間が POI 滞在時間と重なってはならないという条件はあるが，S 区間の最大値は POI 開始条件に含まれる必要がないし，E 区間の最小値は POI 終了条件に含まれる必要がない．これは，たとえば，POI がレストランとなるような応用において自然な仮定である．

ユーザ条件を細かく定義しているのは，できるだけ一般的な状況に対応するためであり，それらには実用上対応する意味がある．仕事終わりに友人と宴会をする場面で図 1 の値を例にして，以下に説明する（以降の説明でも POI への滞在を宴会の比喩で説明することがある）．

- 出発点条件の最小値：仕事が 17 時までなので，それ以前は出発できない．
- 出発点条件の最大値：18 時に会社が閉まるので，その前に出発する．
- POI 開始条件の最小値：友人が 19 時前に来られないので，それ以降に宴会を開始する．
- POI 開始条件の最大値：宴会を遅くとも 20 時より前に始めたい．
- POI 終了条件の最小値：21 時まではゆっくり宴会をしたい．
- POI 終了条件の最大値：22 時以降は太るので宴会を終わらせる．
- 到着条件の最小値：23 時にならないと家のカギが開かない．
- 到着条件の最大値：門限が 24 時である．
- 滞在長条件の最小値：2 時間は最低ゆっくり宴会をしたい．
- 滞在長条件の最大値：長く滞在すると腰が痛くなり 2 時間が限界．

また，チケット購入のように，窓口に滑り込みで間に合えばよいという場合，滞在長条件を 0 に設定すればよい．

今後の説明のために，ユーザ条件と解の区間の最大値と最小値に名前を付ける（図 1）．

- 出発点条件 $= [start\_min, start\_max]$
- POI 開始条件 $= [poi\_start\_min, poi\_start\_max]$
- POI 終了条件 $= [poi\_end\_min, poi\_end\_max]$

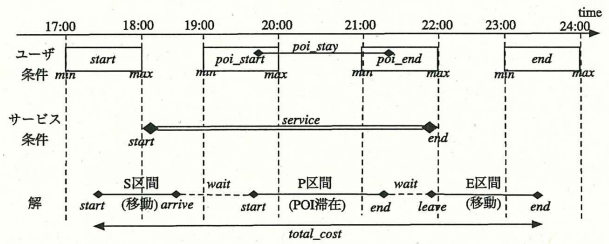


図 1 タイムセール寄り道探索の解

Fig. 1 Answer of time-sale trip planning search.

- 滞在長条件の最小値 $= poi\_stay\_min$
- 滞在長条件の最大値 $= poi\_stay\_max$
- S 区間 $= [total\_start, poi\_arrive]$
- P 区間 $= [poi\_stay\_start, poi\_stay\_end]$
- E 区間 $= [poi\_leave, total\_end]$

総所要時間を $total\_cost$，出発点から POI までの時間を $cost\_s$，POI から到着点までの時間を $cost\_e$ と表すこととする．以下の自明な関係が成り立つ．

- $total\_cost = total\_end - total\_start$
- $poi\_arrive = total\_start + cost\_s$
- $total\_end = poi\_leave + cost\_e$

4. タイムセール寄り道探索の基本導出法

4.1 逐次拡大法による寄り道探索

基本導出法の基となる寄り道探索のための逐次拡大法（大沢ら [6]）について説明する．上位 k 個の解を求める逐次拡大法の基本的な手順を図 2 に示す．ダイクストラ法に相当する部分は簡略化している．出発点を S・到着点を E とする．基本的なアイディアは，出発点側（S 側）・到着点側（E 側）の両方からダイクストラ法を実行することである．指定したカテゴリのノードに到達したら，そのノード（これが見つかった候補 POI．複数ありうる）までのルートを候補テーブルに登録し，候補テーブルの中で，候補

```
while 展開すべきノード集合が空ではない do
  N_S ← S 側の次の最小コストノード（ダイクストラ法）
  N_E ← E 側の次の最小コストノード（ダイクストラ法）
  if 終了条件を満たす then
    return 結果テーブル
  end if
  if N_S がユーザ指定カテゴリを持つノード then
    候補テーブルに S → N_S のルートを登録
  end if
  if N_E がユーザ指定カテゴリを持つノード then
    候補テーブルに N_E → E のルートを登録
  end if
  if 候補テーブルに両側から同じ POI に到着するルートがある then
    見つかった S 側・E 側のルートをつなげて，結果テーブルに登録
    候補テーブルから，見つかった S 側・E 側のルートを削除
  end if
  結果テーブルを全体コストでソートし，上位 k 個のみを保持
  コストテーブルの更新（ダイクストラ法）
end while
```

図 2 逐次拡大法のアルゴリズム（概要）

Fig. 2 Overview of incremental algorithm.

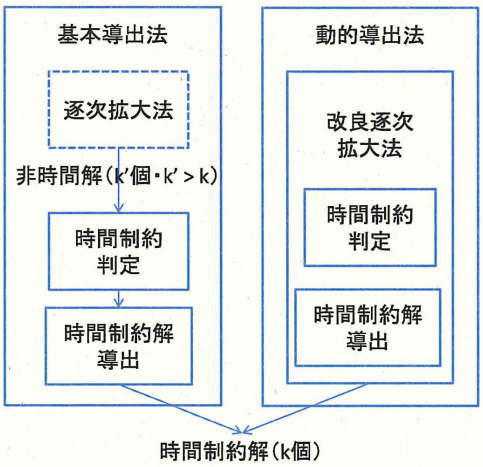


図 3 基本導出法と動的導出法の概要
Fig. 3 Overview of basic method and dynamic method.

POI に両側から到達するルートが登録されたら寄り道が完成したことになるので，それをつなげて結果テーブルに登録する．

繰返しの終了条件であるが，通常のダイクストラ法ならば，到着点のコスト決定時点でよいが，寄り道探索では到着点ではなく POI を探すため，以下の条件とする必要がある．結果テーブルを RT，候補テーブルを CT，ルート R の全体コストを $cost(R)$，出発点（到着点）から POI までのコストを $cost\_s(R)$（$cost\_e(R)$），探索において次に展開するノードのコストを $cost\_s\_current$（S 側），$cost\_e\_current$（E 側）とするとき，終了判定条件は，k 個の解が求められ，かつ以下の 2 つが両方成り立つことである．この式はよりコストの小さいルートが今後得られる可能性がないことを意味している．

$$max_{R \in RT}(cost(R)) \leq min_{R \in CT}(cost\_s(R)) + cost\_e\_current$$

$$max_{R \in RT}(cost(R)) \leq min_{R \in CT}(cost\_e(R)) + cost\_s\_current$$

ただし，POI 密度（全体ノード数に対する POI 数の割合）が低い場合，k 個の解が見つからないと，上記条件だけでは終了せずに延々と遠回りして解を探してしまう．よって，応用目的にあった現実的な値（例：レストラン探索なら 180 分以内）で上限を設けることも必要になる．

4.2 タイムセール寄り道探索解の基本導出法

基本導出法の概要を図 3 の左側に示す．まず既存の逐次拡大法を用いて，時間制約のない解を（最終的に求める個数よりも多めに）求める（非時間解と呼ぶ）．そして，非時間解が時間制約を満たすかを判定し，最終的な解（時間制約解と呼ぶ）を導出する．この方法のメリットは既存の寄り道探索手法をそのまま利用できることにある．一方，問題点は，非時間解を多めに求めたとして，時間制約解がその中に含まれる保証が必ずしもないことである．たとえば，POI 密度が高いが時間制約を満たす POI が少ない場合，逐次拡大法は近くにある POI ばかりを非時間解として求めてしまって，与えられた時間制約を満たす POI に到達しない可能性がある．よって，非時間解は多めに求めておく必要があるが，その個数については 6 章で評価している．

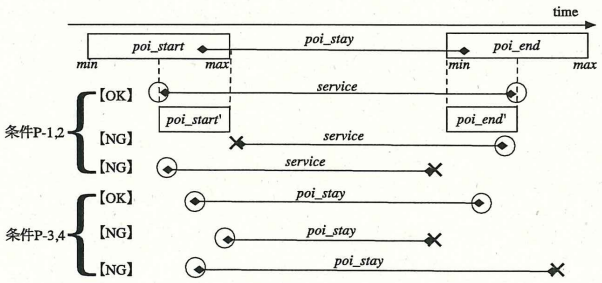


図 4 P 区間の制約チェック
Fig. 4 Constraint check of P-interval.

4.3 基本導出法における時間制約の判定

非時間解が時間制約を満たすかの判定について述べる．3 つという限られた時間区間の判定なので，(a) 単独区間の制約判定，(b) 隣接 2 区間の制約判定，(c) 全体（3 区間）の制約判定，を数直線上でチェックする初等的な方法でよい．また，方針としては，最初にサービス条件と POI 開始条件と POI 終了条件の制約をチェックする．それが満たされるのであれば，それらの交わりを新たな POI 開始条件と POI 終了条件としてその後のチェックに進む．このことで，実質的にサービス条件のチェックが終了するため，その後，サービス条件を考慮する必要がなくなる．つまり最初に P 区間単独の制約判定を行うことになる．

P 区間に関するチェックの例を図 4 に示す．最初に，POI 開始・終了条件とサービス条件に矛盾がないことをチェックする．POI への滞在開始時刻と滞在終了時刻はサービス時間に含まれる必要がある．以下に条件を示す．

- 条件 P-1：$service\_start \leq poi\_start\_max$
- 条件 P-2：$poi\_end\_min \leq service\_end$

次に，POI 開始・終了条件を補正する．これはサービス条件を POI 開始・終了条件に反映させる処理である．POI 開始・終了条件とサービス条件で交わりをとり，それを補正された POI 開始条件・終了条件として今後の処理で使用する（図 4 の poi_start' と poi_end'）．

次に，補正された POI 開始・終了条件と POI 滞在時間に矛盾がないことをチェックする．図 4 に示すように，滞在時間が POI 開始・終了条件に対して短い場合と長い場合は矛盾である．制約を満たす条件を以下に示す．条件 P-3 が POI 滞在が短い場合の排除，条件 P-4 が POI 滞在が長い場合の排除である．これで P 区間単独としての制約チェッ

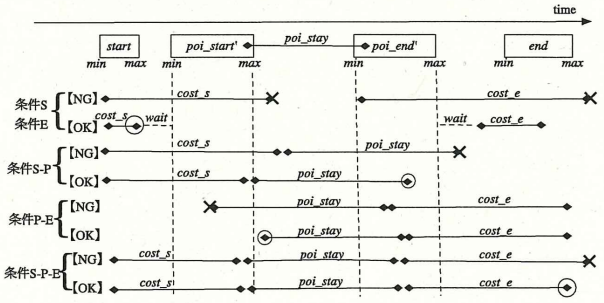


図 5 S 区間と E 区間の制約チェック
Fig. 5 Constraint chek of S-interval and E-interval.

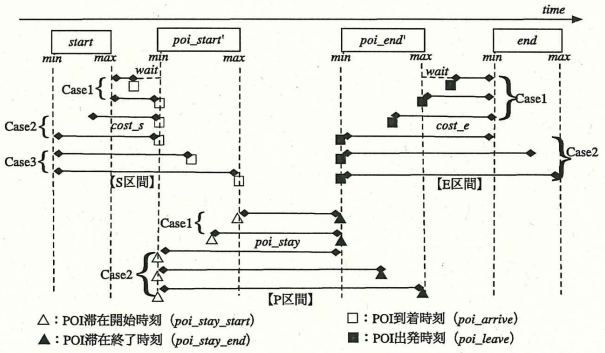


図 6 個別値の決定イメージ
Fig. 6 Overview of each value determination.

クが完了する．

- 条件 P-3：$poi\_end'\_min < poi\_start'\_max + poi\_stay\_min$
- 条件 P-4：$poi\_start'\_min + poi\_stay\_max \leq poi\_end'\_max$

次に，S 区間と E 区間についての制約チェックを行う．非時間解で求まった出発点から POI までの時間（$cost\_s$）と POI から到着点までの時間（$cost\_e$）を利用する．**図 5** にチェックの例を示す．まず，S 区間と E 区間の単独のチェックであるが，待ちを許容するから小さい分には制約はない（POI 開始条件（終了条件）内に到着（出発）する必要はない）ので，大きい場合のチェックのみでよい．S 区間は POI 開始条件を超えなければよく，E 区間は POI 終了条件を超えなければよい．以下に条件を示す．

- 条件 S：$start\_min + cost\_s < poi\_start'\_max$
- 条件 E：$poi\_end'\_min + cost\_e < end\_max$

次に，隣接 2 区間の制約判定であるが，それぞれ 2 つをつなげた区間が大きすぎて，POI 滞在終了・開始を超えるということがなければよい（小さい場合については，単独区間の制約のみでかまわない）．以下に条件を示す．

- 条件 S-E：$start\_min + cost\_s + poi\_stay\_min \leq poi\_end'\_max$
- 条件 P-E：$poi\_start'\_min \leq end\_max - (cost\_e + poi\_stay\_min)$

最後に 3 区間全体の整合であるが，出発から到着までのトータルの長さが区間の最大幅を超えなければよい．以下に条件を示す．

- 条件 S-P-E：$cost\_s + poi\_stay\_min + cost\_e < end\_max - start\_min$

これで制約条件のチェックは完了である．

4.4 基本導出法における時間制約解の導出

次に，時間制約解の導出について説明する．前節のチェックによって，「時間（の長さ）」について矛盾するような条件は排除されているが，これまで与えられた制約だけでは，いつ出発し，POI に滞在し，目的地に到着するという「時刻」は一意には決定されない（総所要時間は一意に決まらない）．時刻を決定するためには，スケジュールの戦略を前提としておく必要がある．ここでは，代表的な「できるだけ待ちをなくし，かつ早く行動する」という戦略に基づく方法を提示する．戦略の特徴を以下にあげる．

- できるだけ早く POI 滞在を開始する．
- できるだけ早く帰宅しようとする．
- 待ち時間はできるだけ発生させない．
- POI でのサービスに間に合う範囲で，出発点の出発時間を遅らせる．

この戦略は単純化されており，現実の適用領域によっては適さない場合もあるが，それは下記に述べる方法を基本として修正すれば実現できる．

以下の導出では単純化のため P 区間の長さは固定であると仮定する（この値を poi_stay とする）．

時間制約解の導出は 2 つのフェーズによって構成される．

- 個別値決定フェーズ
- 調整フェーズ

個別値決定フェーズとは，全体としてのスケジュールの整合性は無視し，3 つの区間の位置をそれぞれ戦略に合うように（この場合はできるだけ早めに行動するように）決定する．調整フェーズでは，前フェーズで決まった区間が重ならないように調整する．

図 6 に個別値決定フェーズのイメージを示し，決定するための場合分け条件を**表 2** に示す．この表の条件ですべての値が決定される．この表に入らないような区間が長すぎるケース，短すぎるケースは事前に除外されている．

S 区間は，$cost\_s$ の値が小さいときは，許容されるぎりぎり（$start\_max$）に出発し，POI 到着後宴会開始まで待つことになる（Case1）．$cost\_s$ がだんだん大きくなり待ちが短くなり，poi_arrive が poi_start_min に到達すると，次に $total\_start$ が前倒しされる（Case2）．$total\_start$ が $start\_min$ に到達した時点で，poi_arrive が後ろにずれていく（Case3）．$cost\_s$ の最大値は $poi\_start\_max - start\_min$ である．

P 区間は，POI 滞在時間（poi_stay）が小さいときは，許容されるぎりぎり（poi_end_min）に終了するように宴

表 2 個別値の決定
Table 2 Each value determination.

場合 (Case)		条件	変数	値	備考 (意味解釈)
S 区間	1	*cost_s =< poi_start_min - start_max*	*total_start*	*start_max*	往路が短く待ち発生
	2	*poi_start_min - start_max < cost_s* *cost_s <= poi_start_min - start_min*	*total_start*	*poi_start_min - cost_s*	待たずに最も早く宴会開始
	3	*poi_start_min - start_min < cost_s* *cost_s <= poi_start_max - start_min*	*total_start*	*start_min*	往路が長く宴会が後ろにずれる
P 区間	1	*poi_end_min - poi_start_max <= poi_stay* *poi_stay < poi_end_min - poi_start_min*	*poi_stay_start*	*poi_end_min - poi_stay*	宴会は最も早く終る
	2	*poi_end_min - poi_start_min <= poi_stay* *poi_stay <= poi_end_max - poi_start_min*	*poi_stay_start*	*poi_start_min*	宴会が長いため終了が後ろにずれる
E 区間	1	*cost_e =< end_min - poi_end_min*	*poi_leave*	*end_min - cost_s*	最も帰宅が早い
	2	*end_min - poi_end_min < cost_e* *cost_e <= end_max - poi_end_min*	*poi_leave*	*poi_end_min*	復路が長く帰宅が後ろにずれる

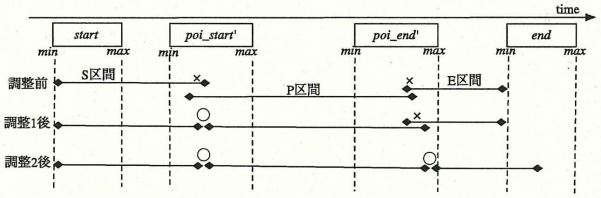


図 7 調整フェーズのイメージ
Fig. 7 Overview of adjustment phase.

会が行われる（Case1）．POI 滞在時間がだんだん大きくなり，*poi_stay_start* が *poi_start_min* に到達すると，次に *poi_stay_end* が後ろにずれる（Case2）．POI 滞在時間の最大値は $poi\_end\_max - poi\_start\_min$ である．

E 区間は，*cost_e* の値が小さいときは，許容されるぎりぎり（*end_min*）に到着するように POI を出発する．宴会終了後 POI 出発まで待つことになる（Case1）．*cost_e* がだんだん大きくなり待ちが短くなり，*poi_leave* が *poi_end_min* に到達すると，次に *total_end* が後ろにずれる（Case2）．*cost_e* の最大値は $end\_max - poi\_end\_min$ である．

次に調整フェーズについて述べる．上記パターンの組合せによって全体が決定されるが，調整が必要な場合として，一般的には以下の 2 つがある．

- 各区間の間に空きがある場合
- 各区間に重なりがある場合

しかし，POI 滞在時間を固定としたために，空きがある場合は，それは待ち時間となり，調整の余地はない．重なりがある場合の解決方法であるが，イメージを図 7 に示す．個別値決定では可能なかぎり，前倒しになるようにスケジュールが決定されているので，重なりが発生した場合は，後の区間を後ろにずらせばよい．S 区間と P 区間が重なったら，P 区間を後ろにずらし，さらに P 区間と E 区間が重なったら E 区間を後ろにずらす．その結果，*total_end* が *end_max* を超えるケースは事前に除外されているため正しい解が得られる．

5. タイムセール寄り道探索解の動的導出法

動的導出法は，時間制約条件を積極的に利用して，時間制約解を探索中に完成させ，事前に候補を絞り込み性能向上をはかる方式である．図 3 の右側にイメージを示しているが，基本導出法で用いた時間制約判定と時間制約解導出を逐次拡大法の中に取り込んでいる．単純に取り込むだけではなく，できるだけ早めに絞り込みを行うための修正を行っている．

基本導出法における時間制約判定は，最初に P 区間のチェックを行い，POI 開始・終了条件の補正を行ったために，このままではすべての条件が POI 確定後ではないとチェックできない．しかし，条件 S，条件 E，条件 S-P，条件 P-E は，補正前の POI 開始・終了条件を用いてチェックすることができ，POI 確定前にチェックできる条件である．以下に条件を示す（それぞれ元の条件を補正前に変更したものである）．

- 条件 S'：$start\_min + cost\_s \leq poi\_start\_max$
- 条件 E'：$poi\_end\_min + cost\_e \leq end\_max$
- 条件 S-P'：$start\_min + cost\_s + poi\_stay\_min \leq poi\_end\_max$
- 条件 P-E'：$poi\_start\_min \leq end\_max - (cost\_e + poi\_stay\_min)$

この 4 条件を含んだ 13 条件をできるだけ早い段階でチェックすることにより性能向上をはかる．その処理フローを，図 8 に示す（図の (b) は (a) における「S 側（E 側）で確定したノードが POI かチェック」の中のフローである）．元となっている逐次拡大法に対する修正点を，図中で二重線で示している．以下の 3 つのステップで候補を絞り込む．それぞれでチェックできる条件を括弧内に書く．

- Step1：ノードへの最短距離が確定する時点（条件 S', E', S-P', P-E'）
- Step2：片側から POI に到達する時点（条件 S，E，P-1〜4，S-P，P-E）
- Step3：POI への両側からのパスが見つかる時点（条

件 S-P-E）

Step1 のチェックは，探索範囲の限定に効果がある．Step2 のチェックは，解の候補テーブルを小さくする効果がある．Step3 のチェックは，最終的に全体としての制約を満たすため必要なチェックである．条件 S’，E’，S-P’，P-E’ は，POI に到達する前にチェックできるので，Step1 でのチェックが可能である．チェックの結果 NG になった場合は，そのノードに至るコストを無限大に設定し，そのノードの先を展開しないようにする．一方，条件 S，E，P-1〜4，S-P，P-E は，POI に到達しないとチェックができないが，両側からのコストを要求していないので，Step2 でのチェックが可能である．チェックの結果 NG になった場合は，候補テーブルへの追加を行わない．条件 S-P-E は両側からのコストが必要なので，Step3 でのチェックが必要である．チェックの結果 NG になった場合は結果テーブルへの追加を行わない．

チェックが終わってから，S，P，E 区間を導出するが，これは基本導出法と同じ方法である．そこで求まった総所要時間によって，結果テーブルのソートを行う．

また，終了条件において，POI 滞在時間を考慮する必要があるため，4.1 節で述べた逐次拡大法における終了条件を以下のように修正する必要がある．

$$max_{R \in RT}(cost(R)) \leq min_{R \in CT}(cost\_s(R)) + cost\_e\_current + poi\_stay\_min$$

$$max_{R \in RT}(cost(R)) \leq min_{R \in CT}(cost\_e(R)) + cost\_s\_current + poi\_stay\_min$$

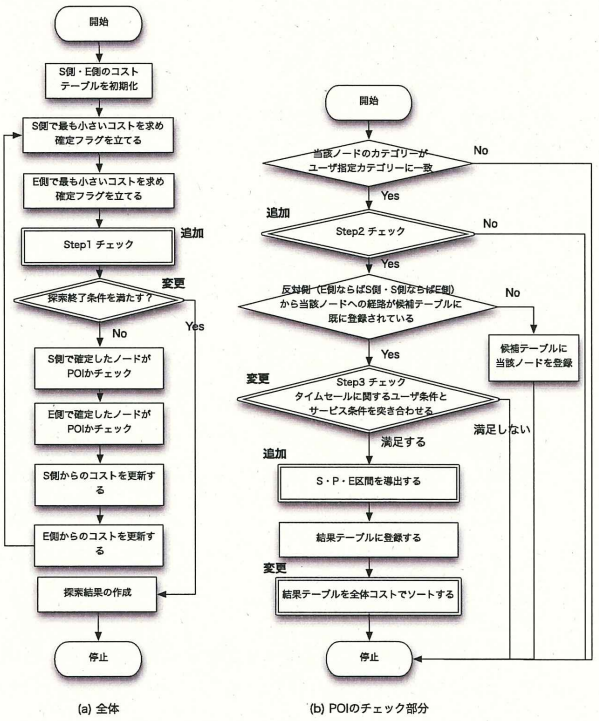


図 8 動的導出法のフロー図

Fig. 8 Flow of dynamic method.

6. システムの実装と評価

前章までに述べた基本導出法，動的導出法を実装し評価した．グラフ情報を実装するために，グラフデータベースである Neo4j [16] を利用した．首都圏の実際の鉄道網の情報と，POI 情報をグラフデータベースとして構築した（ノード数 64 万件，エッジ数 60 万件）．グラフデータの例を図 9 に示す．鉄道網に関しては，駅をノード，路線をエッジとして表現し，POI 情報はノードとして表現し，駅と POI の間は徒歩による移動時間を入れてエッジとして表現している．また，今回の実験システムでは，サービス情報と時間情報もグラフデータベース内のノードとして構築した．サービス情報と時間情報に対するエッジは，鉄道網と POI 情報のエッジと種別を区別することによって，グラフ探索の範囲から除外している．サービス情報と時間情報はできるだけ汎用的になるように設計している．1 つの POI が複数のサービス（例：ランチサービス，宴会サービス）を持つ場合，それは別ノードとして表現する．時間情報は 1 つの区間に対応しており，1 つのサービスに対して，複数区間の営業時間がある場合（例：ランチ後に休憩あり），時間情報のノードを複数持たせることによって表現することができる．時間情報にはサービス実施，非実施の識別子があり，サービスを実施している区間とサービスを実施していない区間という表現もできる．

このグラフデータベースに対して，タイムセール寄り道探索の基本導出法，動的導出法を実装し，Web からもアクセス可能な，タイムセール寄り道探索の実験システムを構築した（実験環境：Apple MacPro（Xeon 2.8 GHz × 2），メモリ 12 GB，SSD128 GB，Mac OS X 10.6.7．プログラミング言語：Java（JDK 6））．その画面例を図 10 に示す．ユーザは出発地，到着地，寄り道先のカテゴリ，サービス

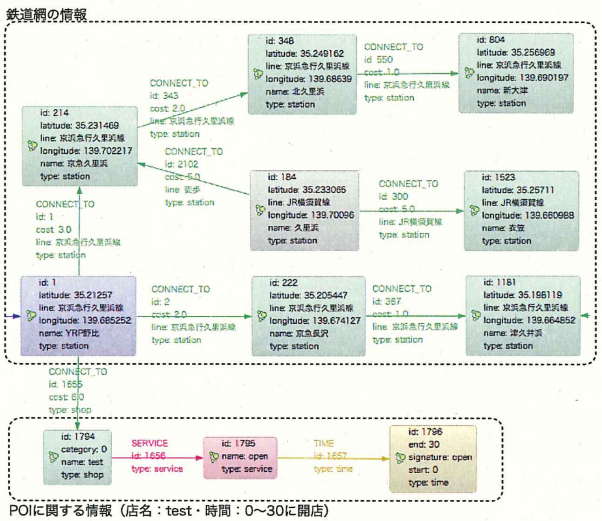


図 9 グラフデータベースのデータ例

Fig. 9 Sample of graph database data.

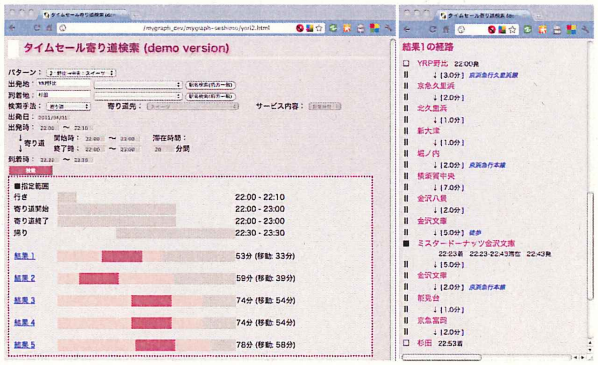


図 10 寄り道探索システムの画面
Fig. 10 Sample window of trip planning search system.

内容，ユーザ条件となる時間制約を指定して探索実行すると，探索結果のスケジュールとルートを返却する．画面は実験目的であるので，すべての時間制約を入力可能にしているが，実際のアプリケーションでは，時間制約をスケジュールシステムから取得したり，個人ごとに事前設定したりするやり方も考えられる．

まず，基本導出法において，中間解として生成される非時間解の個数をどの程度にすべきかを見積もった．データとしては上記の鉄道網データに加えて，サービス密度（交通網のノード数に対して，指定されるサービス内容を実施している POI の数の割合）を変化させ，評価用に自動生成した POI データを用いている（ノード数とエッジ数はともに約 4,000 件）．また，POI は各駅に隣接して 1 つずつ設定した．よって，サービス密度が 100%とは探索するすべての駅ごとにサービスが実行されているという意味となる．

我々は，本論文の手法の適用範囲を，サービス密度が 20%以下の比較的サービス密度が低い場合を想定し，評価した．サービス密度が高い場合，本論文で述べたような手法を用いずに，出発点と到着点の間の最短経路を求め，その経路上にある POI を列挙する簡易な手法でも，ある程度実用的な解を求められることが，その理由である（POI の分布が偏っていれば，正解を見つけられない場合はある）．しかし，サービス密度が 20%の場合とは，今回の評価データである鉄道網であれば，5 駅のうち 1 つでサービスを実施することに相当し，道路網においては，エッジの長さが信号間の距離 200 m 程度であると想定すれば，約 1 km ごとにサービスがあることに相当する（エッジが直線的に連続する場合）．これは，サービスへの適用範囲としては十分に広いと考えた．実際の実用的なサービスにおけるサービス密度の調査については今後の課題である．

まず，動的導出法によって正解となる 10 個の時間制約解を求めておいて，非時間解の個数を変化させて，同じ条件で基本導出法を実行したときに正解となる時間制約解を 10 個中何個導出できるかを調べた．評価結果を図 **11** に示す．サービス密度が低くなる場合，非時間解を多く生成

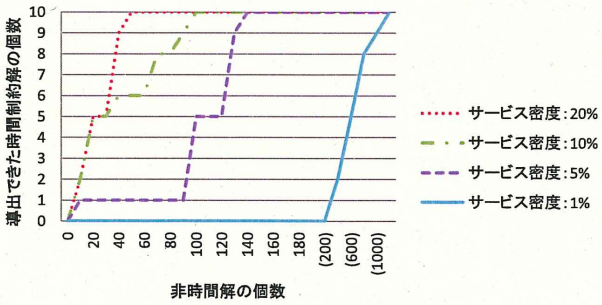


図 11 基本導出法における非時間解の個数と時間制約解の個数の関係
Fig. 11 Relationship of no constrained answers and time constrained answers.

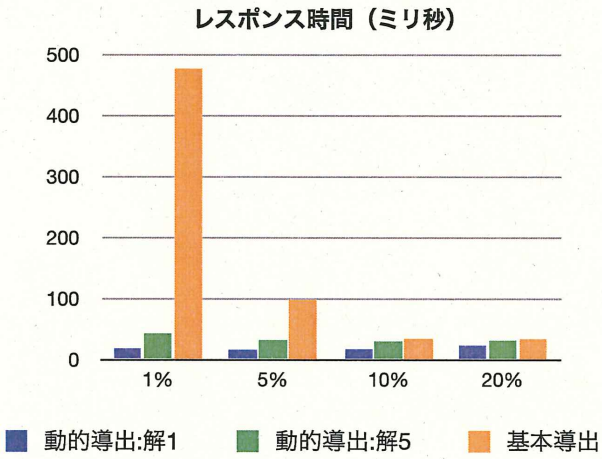


図 12 基本導出法と動的導出法の比較（レスポンス時間）
Fig. 12 Evaluation of response time.

しないと十分な時間制約解を求めることができない．以降の評価では，最終的に返却する解の個数を 1，5 の 2 通りについて評価するために，5 個の正解を得ることができるように，非時間解の個数を 1%：500，5%：100，10%：20，20%：20，と設定した．

次に，基本導出法と動的導出法の探索を実行し，展開された総ノード数，候補テーブルの要素の最大個数，レスポンスタイムを測定した．ユーザの利用シーンとして，以下を想定した．

- 移動途中の 1 時間の空き時間に，タイムセールで安くなったケーキをお土産として買う．
- 買い物時間は 10 分．
- タイムセールの時間は 1 時間．

この利用シーンに相当する時間制約を与え探索を実行し，測定した結果を図 **12**，図 **13**，図 **14** に示す（出発点，到着点を 2,000 組ランダムに選び，その中で解が見つかったものの平均値を示している）．時間制約解の個数は 1，5 の 2 通り測定したが，基本導出法についてはどちらの結果もほぼ同じである．特にサービス密度が低い 1%，5%の場合に，動的導出法の結果がすぐれている．たとえば，サービス密度が 1%の場合は，動的導出法を用いることにより，総

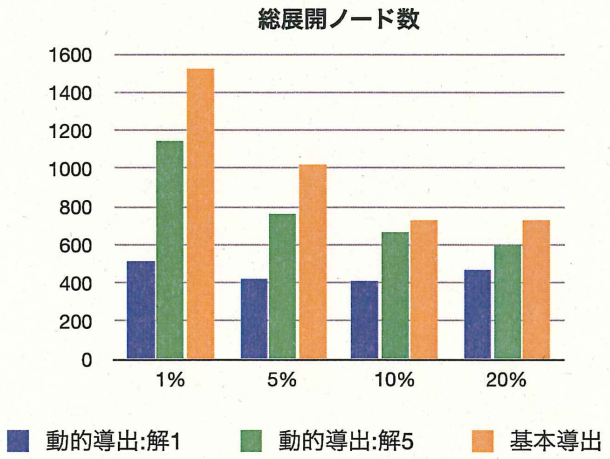


図 13 基本導出法と動的導出法の比較（総展開ノード数）
Fig. 13 Evaluation of number of expanded nodes.

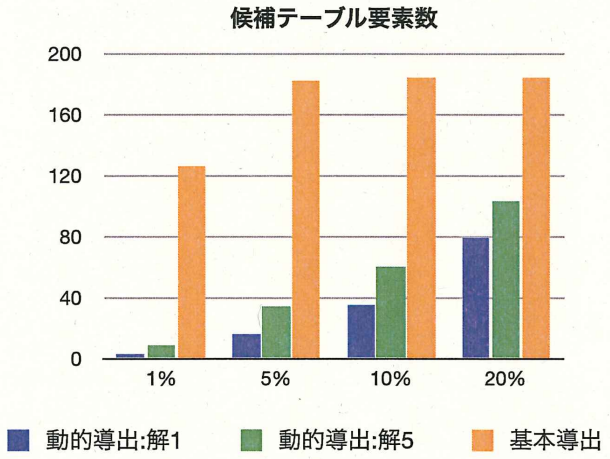


図 14 基本導出法と動的導出法の比較（候補テーブル要素数）
Fig. 14 Evaluation of number of candidate table elements.

展開ノード数が 34～75%に，候補テーブルの要素数が 2～7%に減り，レスポンスタイムが 11～24 倍に改善されている．特に時間制約解が 1 つの場合に改善効果が大きい．また，図 13，図 14 から，総展開ノード数と候補テーブル要素数が動的導出で大幅に削減されており，性能向上に貢献していると推測される．また，サービス密度が高い 20%の場合には，動的導出法の優位性は小さくなるものの，そのオーバヘッドは大きくないことが分かる．今回の測定では，鉄道網というグラフとしては小規模な例で評価したが，道路網等のより大きいデータになれば，この差は広がるため，動的導出法は大規模グラフデータベースに対してはより有効であると考えられる．

7. おわりに

本論文では，タイムセール寄り道探索の考え方を提案し，その実現方法として基本導出法と動的導出法を明らかにした．提案方式をグラフデータベースを利用して実装，評価し，動的導出法がサービス密度が低い場合に特に有効であることを示した．今後の技術的な課題としては，複数の POI への拡張がある．ダイクストラ法を基本とする本手法では複数 POI へ拡張する場合，中間解における組合せが巨大になるという問題が発生する．よって，本手法のように最適解を求めることが難しいため，非最適解を精度，効率良く求める手法が必要になる．また，本技術を適用する観点では，本論文では鉄道網をベースとした典型的な利用シーンを想定して評価したが，本手法は汎用的であるため，より広い利用シーンで利用可能である．タイムセールというと，現状だと夕方限定ということが多いかもしれないが，今後，twitter 等を利用して積極的な時間限定セールのマーケティングを行うとすると，本技術との組合せで新しいサービスが実現できる可能性もある．今後，サービスイメージや利用シーンをさらに検討し，その要求条件に沿ったモデルによる評価が課題となる．

参考文献

[1] Dijkstra, E.W.: A note on two problems in connexion with graphs, *Numerische Mathematik*, Vol.1, pp.269–271 (1959).
[2] Hart, P.E., Nilsson, N.J. and Raphael, B.: A Formal Basis for the Heuristic Determination of Minimum Cost Paths, *IEEE Trans. on Systems Science and Cybernetics*, Vol.SSC4, No.2, pp.100–107 (1968).
[3] 五十嵐健夫：データ構造とアルゴリズム，数理工学社 (2007).
[4] 駅前探検倶楽部，入手先 ⟨http://ekitan.com/⟩（参照 2011-05-23）.
[5] 食べログ，入手先 ⟨http://tabelog.com/⟩（参照 2011-05-23）.
[6] 大沢 裕，藤野和久：前処理を必要としない道路ネットワーク上での最短寄り道経路探索アルゴリズム，電子情報通信学会論文誌 D，Vol.J93-D，No.3，pp.203–210（オンライン）(2010)，入手先 ⟨http://search.ieice.org/⟩.
[7] Sharifzadeh, M., Kolahdouzan, M. and Shahabi, C.: The optimal sequenced route query, *VLDB J.*, Vol.17, pp.765–787 (2008).
[8] Li, F., Cheng, D., Hadjieleftheriou, M., Kollios, G. and Teng, S.-H.: On trip planning queries in spatial databases, *Proc. SSTD 2005*, pp.273–290 (2005).
[9] Papadias, D., Zhang, J., Mamoulis, N. and Tao, Y.: Query processing in spatial network databases, *Proc. 29th VLDB*, pp.790–801 (2003).
[10] 蒲原智也，上島紳一：道路網応用のための空間索引木の提案と最短経路探査への応用，情報処理学会論文誌 データベース，Vol.2，No.2，pp.10–28（オンライン）(2009)，入手先 ⟨http://www.bookpark.ne.jp/ipsj/⟩.
[11] Tansel, A.U., Clifford, J. and Gadia, S.: *Temporal Databases: Theory, Design and Implementation*, Benjamin-Cummings (1993).
[12] Norvag, K.: Temporal Query Operators in XML Databases, *Proc. ACM Symposium on Applied Computing 2002*, pp.402–406 (2002).
[13] Kleinberg, J. and Tardos, E.：アルゴリズムデザイン，共立出版 (2008).
[14] 松田三恵子，杉山博史，土井美和子：歩行者の経路への嗜好を反映した経路生成，電子情報通信学会論文誌 A，Vol.J87-A，No.1，pp.132–139（オンライン）(2004)，入手先 ⟨http://search.ieice.org/⟩.

[15] Cheng, J., Ke, Y. and Ng, W.: Efficient Query Processing on Graph Databases, *ACM Trans. Database Systems*, Vol.34, No.1, pp.1–48 (2009).
[16] Neo4j WEB Page, available from ⟨http://neo4j.org⟩ (accessed 2011-05-23).

鈴木 源吾 （正会員）

1990 年東北大学大学院理学研究科修士課程修了．同年日本電信電話株式会社入社．データベース設計，マルチデータベースシステム，XML データ管理，グラフデータ管理等の研究開発に従事．電子情報通信学会，日本データベース学会各会員．

榎本 俊文 （正会員）

1994 年大阪大学大学院基礎工学研究科修士課程修了．同年日本電信電話株式会社入社．エージェント技術，情報検索，XML データ管理等の研究開発に従事．電子情報通信学会，人工知能学会各会員．

小林 伸幸 （正会員）

1989 年京都大学大学院応用システム科学研究科修士課程修了．同年日本電信電話株式会社入社．データベース管理システム，XML データ管理等の研究開発に従事．電子情報通信学会，日本データベース学会各会員．

山室 雅司 （正会員）

1987 年早稲田大学大学院理工学研究科修士課程修了．同年日本電信電話株式会社入社．1990 年コロンビア大学大学院電気工学研究専攻修士課程修了．以来ネットワーク設計法，データベース設計・可視化，マルチメディア情報検索，データストリーム管理，XML データ管理，グラフデータ管理の研究開発に従事．博士（工学）．1994 年電子情報通信学会学術奨励賞．電子情報通信学会シニア会員，日本ソフトウェア科学会，日本データベース学会，IEEE-CS 各会員．本会マルチメディア通信と分散処理研究会幹事，本会財務理事．

鬼塚 真 （正会員）

1991 年東京工業大学工学部情報工学科卒業．同年日本電信電話株式会社入社．2000～2001 年ワシントン州立大学客員研究員．博士（工学）．ACM，電子情報通信学会，日本データベース学会各会員．